Department of Mathematics

Numerical Solution of Partial Differential Equations

November 30, 2005

Contents

Preface

These Lecture Notes draw on lecture notes supplied by Dr D M Haughton, Department of Mathematics, The University of Glasgow, and have been compiled by the author for the undergraduate course *Numerical Solution of Partial Differential Equations* given in the Fourth Year of Honours Mathematics at the University of Glasgow. The course comprises 25 lectures.

Feedback by the students, both in terms of information about typographical errors and mistakes as much as suggestions for improvement will be much appreciated.

These Lecture Notes complement the course but **do not define it** in respect of examinable material or course content. Topics in these notes may be replaced/augmented within the lecture environment on a year to year basis.

KAL
Glasgow, October 2003

Chapter 1

Numerical and Mathematical Preliminaries

1.1 Introduction

Only the most elementary ordinary and partial differential equations have closed form solutions, that is, solutions which can be expressed in terms of the fundamental functions of mathematics. Even when closed form solutions are available, the value of such a solution can be questionable. For example, let $\mathcal{A}$ be the interior of the square with vertices at $(1,1)$, $(-1,1)$, $(-1,-1)$ and $(1,-1)$, then it may be demonstrated that the solution of the boundary value problem

$$\frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} = -2\,, \qquad (x,y) \in \mathcal{A}$$

with $\phi = 0$ on $\partial\mathcal{A}$, the boundary of $\mathcal{A}$, has exact solution

$$\phi(x,y) = 1 - y^2 - \frac{32}{\pi^3}\sum_{k=1}^{\infty} \frac{(-1)^k \cosh\Big(\dfrac{(2k+1)\pi x}{2}\Big)\cos\Big(\dfrac{(2k+1)\pi y}{2}\Big)}{(2k+1)^3\cosh\Big(\dfrac{(2k+1)\pi}{2}\Big)}\,.$$

The merit of this exact solution is questionable for three important reasons.

- The evaluation of $\phi(x,y)$ at any given point (x,y) requires the computation of a sum. In this case several hundred terms are required to achieve acceptable numerical accuracy, but often the number of terms can run into many thousands.
- More subtly, the terms in this sum are not trivial to calculate despite the fact that each term is expressed in terms of simple functions. For example, how does one calculate the quotient

$$\frac{\cosh\Big(\dfrac{(2k+1)\pi x}{2}\Big)}{\cosh\Big(\dfrac{(2k+1)\pi}{2}\Big)}$$

when k is large? Certainly one **cannot** compute the numerator and denominator separately and divide! This action will very quickly induce numerical overflow for large k since $\cosh(k\pi x) \to \infty$ as $k \to \infty$ whenever x is non-zero.

- A minor modification to the boundary conditions means a new analytical solution. By contrast, such a change has little impact on the construction of a numerical solution to this PDE.

In conclusion, the commonly held belief that a numerical solution is inferior to an analytical solution is a often false. This example illustrates that in many circumstances the reverse may be the case. The important contributions made by analysis in the theory of ordinary and partial differential equations more often lies in the derivation of conditions to ensure the existence and uniqueness of solutions rather than in their analytical construction. It is only after existence and uniqueness of solution are guaranteed that is makes sense to use a numerical procedure to construct the solution of the ODE or PDE.

1.2 Classification problem

The specification of a problem in terms of a differential equation comes in two parts. First, there is the differential equation itself which may be viewed as an algebraic relationship connecting a function and its derivatives. Second, there are two different types of additional conditions which must be supplied in order to select the desired solution from the many possible solutions. These additional conditions are called *initial conditions* and *boundary conditions*.

Initial conditions are conditions to be satisfied *everywhere* for a fixed value of an independent variable, usually interpreted as time and denoted by t.

Boundary conditions are conditions to be satisfied *everywhere* on the boundary $\partial\mathcal{A}$ of the region $\mathcal{A}$ on which the differential equation is defined.

Different types of PDE require different combinations of initial and boundary conditions, and this in turn means a different numerical procedure. Consider, for example, the family of second order partial differential equations

$$A\,\frac{\partial^2 u}{\partial x^2} + B\,\frac{\partial^2 u}{\partial x}\partial y + C\,\frac{\partial^2 u}{\partial y^2} + D\,\frac{\partial u}{\partial x} + E\,\frac{\partial u}{\partial y} + F\,u = g \tag{1.1}$$

in which $A = A(x, y, u_x, u_y, u), \cdots, F = F(x, y, u_x, u_y, u)$. Equations of this type can be classified in one of three ways depending on the value of $B^2 - 4AC$.

1.2.1 Elliptic equation

Equation (1.1) is of *Elliptic* type whenever $B^2 - 4AC < 0$. Elliptic partial differential equations give rise to "Boundary Value Problems" (BVPs), that is, only boundary conditions must be supplied to

complete the specification of the solution of an elliptic PDE. Elliptic equations arise typically in the formulation of static (time independent) problems in, for example, Potential theory, Fluid Mechanics and Elasticity. Poisson's equation

$$\frac{\partial^2\phi}{\partial x^2}+\frac{\partial^2\phi}{\partial y^2}=g(x,y)\,,\qquad (x,y)\in\mathcal{A} \tag{1.2}$$

is the archetypal elliptic equation. Here $\mathcal{A}$ is a simply connected region with boundary curve $\partial\mathcal{A}$. The task is to find a solution of equation (1.2) satisfying the boundary conditions

$$\left[\begin{array}{rcll} \phi & = & f_1(x,y) & (x,y)\in\mathcal{C}_1 \\ \dfrac{\partial\phi}{\partial n} & = & f_2(x,y) & (x,y)\in\mathcal{C}_2 \\ \dfrac{\partial\phi}{\partial n}+f_3(x,y)\phi & = & f_4(x,y) & (x,y)\in\mathcal{C}_3 \end{array}\right.$$

where $f_1(x,y),\cdots,f_4(x,y)$ are known functions and $\partial\mathcal{A}=\mathcal{C}_1\cup\mathcal{C}_2\cup\mathcal{C}_3$. The case $\mathcal{C}_1=\partial\mathcal{A}$ is the *Dirichlet Problem*, the case $\mathcal{C}_2=\partial\mathcal{A}$ is the *Neumann Problem* and the case $\mathcal{C}_3=\partial\mathcal{A}$ is the *Robin Problem*. It is easy to show that the Dirichlet problem for equation (1.2) has a unique solution but that the solution of the Neumann problem is not unique, and may not even exist.

1.2.2 Parabolic equation

Equation (1.1) is of *Parabolic* type whenever $B^2-4AC=0$. Parabolic partial differential equations give rise to "Initial Boundary Value Problems" (IBVPs), that is, initial conditions are required to start the solution at $t=0$ and thereafter boundary conditions must be supplied to complete the specification of the solution for $t>0$. Parabolic equations arise typically in problems involving heat, magnetic fields and flow processes based on randomness. The Diffusion equation

$$\frac{\partial\phi}{\partial t}=\frac{\partial^2\phi}{\partial x^2}+g(x,t)\,,\qquad (t,x)\in(0,\infty)\times(a,b) \tag{1.3}$$

is the archetypal parabolic equation where $(a,b)\subseteq\mathbb{R}$. A solution of equation (1.3) is sought satisfying the respective initial and boundary conditions

$$\phi(0,x)=\Phi(x)\,,\qquad \left[\begin{array}{rcll} \alpha_a(t)\phi(t,a)+\beta_a(t)\dfrac{\partial\phi}{\partial a} & = & f_1(t) & t>0 \\ \alpha_b(t)\phi(t,b)+\beta_b(t)\dfrac{\partial\phi}{\partial b} & = & f_2(t) & t>0 \end{array}\right.$$

where $\alpha_a(t)$, $\alpha_b(t)$, $\beta_a(t)$, $\beta_b(t)$, $f_1(t)$ and $f_2(t)$ are given functions of t. Again there are well established existence and uniqueness results for parabolic equations of type (1.3).

1.2.3 Hyperbolic equation

Equation (1.1) is of *Hyperbolic* type whenever $B^2-4AC>0$. Hyperbolic partial differential equations give rise to "Initial Value Problems" (IVPs), that is, initial conditions are required to start the solution

at $t = 0$. Hyperbolic equations arise typically in problems involving the propagation of disturbances, for example, in Wave Mechanics. The Wave equation

$$\frac{\partial^2 \phi}{\partial t^2} = \frac{\partial^2 \phi}{\partial x^2} + g(x,t)\,, \qquad (t,x) \in (0,\infty) \times (a,b) \tag{1.4}$$

is the archetypal hyperbolic equation. A solution of equation (1.4) is sought satisfying the initial conditions

$$\phi(0,x) = \Phi(x)\,, \qquad \frac{\partial \phi(0,x)}{\partial t} = V(x)$$

where $\Phi(x)$ and $V(x)$ are given functions of x.

1.3 Solution procedures

Numerical procedures to solve differential equations largely decompose into either techniques which aim to find the value of the solution at a finite number of nodes within the domain of definition of the equation and its boundary, or alternatively, techniques which expand the solution as a sum of basis functions defined on the domain of definition of the equation, and then aim to find the coefficients in this expansion. The algorithms to be examined in these lectures are now described briefly.

1.3.1 Finite differences

Finite difference procedures approximate the derivatives appearing in a PDE by sums and differences of function values at a set of discrete points, usually uniformly spaced with respect to each independent variable. The idea is most effectively illustrated in one dimension. Let x and h be real numbers where h is usually regarded as a small positive *step length*, then

$$\begin{aligned} f'(x) &= \frac{f(x+h) - f(x-h)}{2h} + O(h^2)\,, \\ f''(x) &= \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} + O(h^2) \end{aligned}$$

are two expressions which are often used to compute the first and second order derivatives of f at x from knowledge of $f(x-h)$, $f(x)$ and $f(x+h)$. The motivation for these results is based on the realisation that if f is suitably differentiable in a neighbourhood of x then f has a Taylor Series expansion in the vicinity of x so that

$$\begin{aligned} f(x+h) &= f(x) + hf'(x) + \frac{h^2}{2} f''(x) + \frac{h^3}{6} f'''(x) + O(h^4)\,, \\ f(x-h) &= f(x) - hf'(x) + \frac{h^2}{2} f''(x) - \frac{h^3}{6} f'''(x) + O(h^4)\,. \end{aligned}$$

Trivially,

$$\begin{aligned} f(x+h) - f(x-h) &= 2hf'(x) - \frac{h^3}{3} f'''(x) + O(h^5) \\ f(x+h) - 2f(x) + f(x-h) &= 2h^2 f''(x) + O(h^4) \end{aligned}$$

from which it follows that

$$\begin{aligned} f'(x) &= \frac{f(x+h)-f(x-h)}{2h} + O(h^2) \,, \\ f''(x) &= \frac{f(x+h)-2f(x)+f(x-h)}{h^2} + O(h^2) \,. \end{aligned}$$

These are known as the *central difference* formulae for the first and second derivatives of f. The formulae are "second order" accurate by which we mean that the error incurred by approximating $f'(x)$ or $f''(x)$ by these formulae is proportional to h^2. This error is commonly called the *truncation error* in the sense that it is the mathematical error which is incurred by truncating the Taylor series to obtain the finite difference approximations. In fact, it is possible to find higher order central difference formulae but these will not be used in this lecture course.

Two further formulae are required to estimate right-hand and left-hand derivatives. Such formulae are required for boundary conditions. It can be shown that

$$\begin{aligned} 4f(x+h)-f(x+2h)-3f(x) &= 2hf'(x)+O(h^3) \,, \\ f(x-2h)+3f(x)-4f(x-h) &= 2hf'(x)+O(h^3) \,, \end{aligned}$$

leading to the left-handed and right-handed formulae

$$\begin{aligned} f'(x) &= \frac{4f(x+h)-f(x+2h)-3f(x)}{2h} + O(h^2) \,, \\ f'(x) &= \frac{f(x-2h)+3f(x)-4f(x-h)}{2h} + O(h^2) \,. \end{aligned}$$

Example Use the table

x	0.5	0.6	0.7
$\sin x$	0.47943	0.56464	0.64422

to estimate the first derivative of $\sin x$ at each mesh point. Compare your estimates with the exact answer. Estimate the second derivative of $\sin x$ at $x = 0.6$.

Solution Take $h = 0.1$ then

$$\begin{aligned} f'(0.5) &= \frac{4(0.56464)-3(0.47943)-(0.64422)}{0.2} \approx 0.88052 \quad \text{(Correct ans) } 0.87758 \\ f'(0.6) &= \frac{(0.64422)-(0.47943)}{0.2} \approx 0.82395 \quad \text{(Correct ans) } 0.82533 \\ f'(0.7) &= \frac{3(0.64422)+(0.47943)-4(0.56464)}{0.2} \approx 0.76765 \quad \text{(Correct ans) } 0.76484 \end{aligned}$$

Further

$$f''(0.6) = \frac{(0.64422)-2(0.56464)+(0.47943)}{0.01} \approx -0.56300 \quad \text{(Correct ans) } -0.56464$$

Numerical formulation of finite difference formulae

Let interval $[a, b]$ be subdivided into n smaller intervals of uniform size $h = (b-a)/n$ by the dissection point $x_k = a + kh$, $k = 0, 1, \ldots n$. Let $f_k = f(x_k)$ be the value of f at the point x_k and suppose that $f_0, f_1, \ldots f_n$ are known. The finite difference formulae for the first and second order derivatives of f at an interior point x_k of $[a, b]$ are

$$\begin{aligned} f'(x_k) &= \frac{f_{k+1} - f_{k-1}}{2h} + O(h^2)\,, \\ f''(x_k) &= \frac{f_{k+1} - 2f_k + f_{k-1}}{h^2} + O(h^2)\,. \end{aligned}$$

The corresponding finite difference formulae for the derivative of f at the left hand side and right hand side of the interval $[a, b]$ are

$$\begin{aligned} f'(x_0) &= \frac{4f_1 - f_2 - 3f_0}{2h} + O(h^2)\,, \\ f'(x_n) &= \frac{f_{n-2} + 3f_n - 4f_{n-1}}{2h} + O(h^2)\,. \end{aligned}$$

Example Use the method of finite differences with five nodes to estimate the solution of the ordinary differential equation

$$\frac{d^2u}{dx^2} = x\,, \qquad u(0) = u(1) = 0\,.$$

Solution Let $u_0 = u(0) = 0$, $u_1 = u(1/4)$, $u_2 = u(1/2)$, $u_3 = u(3/4)$ and $u_4 = u(1) = 0$, then the unknown values u_1, u_2 and u_3 are determined by the equations

$$\left.\frac{d^2u}{dx^2}\right|_{x=1/4} = 1/4\,, \qquad \left.\frac{d^2u}{dx^2}\right|_{x=1/2} = 1/2\,, \qquad \left.\frac{d^2u}{dx^2}\right|_{x=3/4} = 3/4\,.$$

With $h = 1/4$, the finite difference formulae give

$$\frac{u_2 - 2u_1 + u_0}{h^2} = 1/4\,, \qquad \frac{u_3 - 2u_2 + u_1}{h^2} = 1/2\,, \qquad \frac{u_4 - 2u_3 + u_2}{h^2} = 3/4\,.$$

Since $u_0 = u_4 = 0$ and $h = 1/4$ then it now follows that

$$\begin{aligned} u_2 - 2u_1 &= 1/64 \\ u_3 - 2u_2 + u_1 &= 1/32 \\ -2u_3 + u_2 &= 3/64. \end{aligned} \qquad \rightarrow \qquad \begin{bmatrix} -2 & 1 & 0 \\ 1 & -2 & 1 \\ 0 & 1 & -2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} 1/64 \\ 1/32 \\ 3/64 \end{bmatrix}.$$

The exact solution of this problem is $u(x) = (x^3 - x)/6$ which in turn gives the exact solutions $u_1 = -5/128$, $u_2 = -1/16$ and $u_3 = -7/128$. In fact, these are also the values taken by the estimated solutions. Why are the numerical solutions exact! The answer lies in the fact that the numerical error in estimating d^2f/dx^2 as a central difference depends on the fourth derivative of f which is zero for the exact solution in this case.

1.3.2 Finite elements

In a finite element procedure the solution is expanded in terms of a set of *basis functions* superimposed on a family of *elements* which in turn span the region in which the solution of the differential equation is sought. The idea is illustrated easily by considering the representation of a function of a single variable.

Let $f(x)$ be a function defined on the interval $[a, b]$ and let $a = x_0 < x_1 < \cdots x_n = b$ be a set of user-specified nodes. In this one-dimensional case, the intervals $[x_0, x_1]$, $[x_1, x_2]$ $\cdots$ $[x_{n-1}, x_n]$ define the elements which span $[a, b]$. These elements are overlayed with a family of user-supplied basis functions. The most common family of basis functions is the set of triangular (or tent-like) basis functions

$$u_k(x) = \begin{cases} \dfrac{x - x_{k-1}}{x_k - x_{k-1}} & x \in [x_{k-1}, x_k] \\ \dfrac{x_{k+1} - x}{x_{k+1} - x_k} & x \in (x_k, x_{k+1}] \\ 0 & \text{otherwise}\,. \end{cases}$$

These are illustrated in Figure 1.1 with $u_k(x)$ highlighted as the shaded triangle.

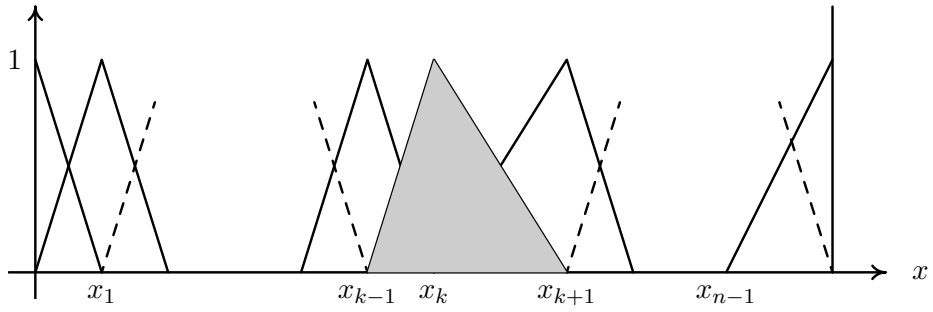


Figure 1.1: Basis functions $u_0(x), u_1(x), \cdots, u_n(x)$ with member $u_k(x)$ shaded.

With respect to these basis functions, the function $f(x)$ is represented by the finite element expansion

$$\widehat{f}(x) = \sum_{k=0}^{n} f_k\, u_k(x)\,.$$

in which $f_0, \cdots, f_n$ is a sequence of coefficients to be determined. It is not immediately obvious how these should be chosen in an optimal way. The *Least Squares* criterion is used frequently for this purpose. The objective of this criterion is to minimise

$$E(f_0, f_1, \cdots, f_n) = \int_0^L \Big[f(x) - \widehat{f}(x)\Big]^2\, dx = \int_0^L \Big[f(x) - \sum_{k=0}^{n} f_k\, u_k(x)\Big]^2\, dx\,,$$

by an appropriate choice of the coefficients $f_0, \cdots, f_n$. The minimum of $E(f_0, f_1, \cdots, f_n)$ is attained when $f_0, \cdots, f_n$ are solutions of the equations

$$\frac{\partial E}{\partial f_j} = -2 \int_0^L \Big[f(x) - \sum_{k=0}^{n} f_k\, u_k(x)\Big] u_j(x)\, dx = 0\,, \qquad j = 0, \cdots, n\,,$$

which in turn asserts that $f_0, \cdots, f_n$ satisfy the $(n+1)$ simultaneous equations

$$\sum_{k=0}^{n} f_k \int_0^L u_j(x)\, u_k(x)\, dx = \int_0^L f(x)\, u_j(x)\, dx\,, \qquad j = 0, \cdots, n\,.$$

Numerical work involving the finite element procedure necessarily requires the evaluation of various integrals with integrands formed from products of basis functions and their derivatives. Triangular basis functions are particularly popular because these basis functions yield integrals that are straight forward to evaluate. However, the use of triangular basis functions is necessarily restricted to low order partial differential equations (2nd order) as will become obvious on further investigation of the approach.

1.3.3 Spectral expansion

The spectral approach to the solution of a differential equation appears superficially similar to the finite element approach in the respect that the solution is expressed as a sum over basis functions. For example, in one dimension the function $f(x)$ is represented by the finite sum

$$\widehat{f}(x) = \sum_{k=0}^{n} f_k\, \phi_k(x)$$

where ϕ_0, $\phi_1, \cdots$ are an infinite family of spectral functions, the first $(n+1)$ of which are used to approximate $f(x)$. The term "spectral" comes from the fact that the basis functions ϕ_k are almost invariably chosen to be the eigenfunctions of a Sturm-Liouville problem. In particular, the members of the family form a *complete* basis for the solution space and are also mutually orthogonal with respect to a suitable weight function. For example, the Fourier series over $[0, L]$ corresponds to the spectral expansion

$$\sum_{k=-n/2}^{n/2-1} c_k\, e^{2\pi i k x/L}\,, \qquad x \in [0, L]\,.$$

Chebyshev polynomials and Legendre polynomials are other common choices of basis functions.

There are various ways to construct a spectral approximation to the solution of a differential equation. For example, one approach requires the spectral solution to minimise some user-supplied criterion, for example, that the weighted least squares estimate of $f(x) - \widehat{f}(x)$ (error between the exact solution and the numerical solution) is minimised. Another approach might force the numerical solution $\widehat{f}(x)$ to satisfy the differential equation identically at a prescribed set of points (*collocation* points) within the domain of definition of f.

1.3.4 Efficacy of numerical algorithms

The efficacy of a numerical algorithm to solve partial differential equations depends on a number of different factors such as the ease of implementation of the algorithm, the numerical accuracy of the

algorithm, the numerical stability of the algorithm, the level of numerical effort required to achieve a prescribed accuracy, the memory overhead of the algorithm and, most importantly, the versatility of the algorithm in terms of its ability to treat a variety of domains of integration.

Of the three approaches outlined in this introduction, the finite difference procedure is the easiest to implement and to analyse mathematically. Next comes the finite element procedure are finally the spectral procedure. The implementation of spectral algorithms requires a significant level of mathematical skill. Of course, the trade-off is that spectral algorithms are much more accurate than either finite difference or finite element algorithms, both of which are of comparable accuracy. The usual analogy is between exponential (spectral) and algebraic (finite difference and finite element) convergence of numerical error to zero as the algorithm is "refined".

Both finite difference and spectral algorithms are best suited to domains with regular boundaries by which we mean boundaries defined by constant coordinate values. By contrast, finite element methods are well suited to regions of integration for which the boundaries are irregular. We have the following rough rule for selection of numerical scheme:-

(a) Choose a finite difference algorithm if the region is regular (in the coordinate sense) and a quick and relatively inaccurate solution is sought.

(b) Choose a spectral algorithm if the region is regular (in the coordinate sense) and an accurate solution is sought.

(c) Choose a finite element algorithm if the region is irregular (in the coordinate sense) irrespective of the numerical accuracy required of the solution.

Exercises

Question 1. Show that there exists η and ξ such that

$$\begin{aligned} f'(x) &= \frac{f(x+h)-f(x-h)}{2h} - \frac{h^2}{6}\, f'''(\eta)\,, \\ f''(x) &= \frac{f(x+h)-2f(x)+f(x-h)}{h^2} - \frac{h^2}{12}\, f^{iv}(\xi)\,. \end{aligned}$$

Question 2. Use the Taylor series expansion of $f(x+h)$ up to and including terms of order h^4 to show that for sufficiently differentiable functions $f(x)$

$$\frac{df}{dx} = \frac{4f(x+h)-f(x+2h)-3f(x)}{2h} + O(h^2)\,.$$

Deduce a similar expression for df/dx in terms of $f(x-h)$ and $f(x-2h)$.

Question 3. Use the Taylor series expansion of $f(x+h)$ to derive the formulae

$$\begin{aligned} \frac{d^3u}{dx^3} &= \frac{u(x+2h)-2u(x+h)+2u(x-h)-u(x-2h)}{2h^3} + O(h^2)\,, \\ \frac{d^4u}{dx^4} &= \frac{u(x+2h)-4u(x+h)+6u(x)-4u(x-h)+u(x-2h)}{h^4} + O(h^2)\,. \end{aligned}$$

Question 4. Construct a finite difference approximation for

$$\psi(x) = \frac{d}{dx}\left(\, a(x)\, \frac{df}{dx}\,\right)$$

that is second order accurate, that is, $\psi(x)$ is estimated to order h^2 in terms of $f(x-h)$, $a(x-h)$, $f(x)$, $a(x)$, $f(x+h)$, and $a(x+h)$. Prove your assertion.

Question 5. Given the value of function f at the points $x-h$ and $x+\theta h$ where $(0<\theta<1)$, find a first order difference formula for the derivative of f at x.

Question 6. Given the value of function f at $x-h$, x and $x+\theta h$ where $(0<\theta<1)$, find a second order difference formula for the derivative of f at x.

Question 7. Given the value of function f at the points $x-2h$, $x-h$, x and $x+\theta h$ where $(0<\theta\leq 1)$, derive the finite difference formula

$$f''(x) = \frac{6f_1 + (\theta-3)(\theta+1)(\theta+2)f_0 - 2\theta(\theta^2-4)f_{-1} + \theta(\theta^2-1)f_{-2}}{\theta(\theta+1)(\theta+2)h^2} + O(h^2).$$

Question 8. Derive the higher order central difference formulae

$$\begin{aligned} \frac{du}{dx} &= \frac{-u(x+2h)+8u(x+h)-8u(x-h)+u(x-2h)}{12h} + O(h^4)\,, \\ \frac{d^2u}{dx^2} &= \frac{-u(x+2h)+16u(x+h)-30u(x)+16u(x-h)-u(x-2h)}{12h^2} + O(h^4). \end{aligned}$$

Why do you suppose that these are not used instead of the less accurate second order formulae?

Question 9. For the triangular basis functions

$$u_k(x) = \begin{cases} \dfrac{x - x_{k-1}}{x_k - x_{k-1}} & x \in [x_{k-1}, x_k] \\ \dfrac{x_{k+1} - x}{x_{k+1} - x_k} & x \in (x_k, x_{k+1}] \\ 0 & \text{otherwise}\,. \end{cases}$$

in the finite element procedure, compute the values of

$$(a) \quad \int_0^L u_j(x) u_k(x)\, dx\,, \qquad (b) \quad \int_0^L \frac{du_j}{dx}\,\frac{du_k}{dx}\, dx\,,$$

$$(c) \quad \int_0^L u_i(x) u_j(x) u_k(x)\, dx\,, \qquad (d) \quad \int_0^L u_i\,\frac{du_j}{dx}\,\frac{du_k}{dx}\, dx\,.$$

Question 10. Fourier analysis in one dimension represents the solution $f(x)$ of a differential equation by the partial sum

$$\widehat{f}(x) = \sum_{k=-n/2}^{n/2-1} c_k\, e^{2\pi i k x/L}\,, \qquad x \in [0, L]\,.$$

If $f_j = \widehat{f}(jL/n)$, write down the equations connecting $f_0, \cdots, f_{n-1}$ with the coefficients c_k. Demonstrate that these equations have solution

$$c_k = \frac{1}{n} \sum_{j=0}^{n-1} f_j\, e^{-2\pi i j k/n}\,.$$

Chapter 2

Finite Difference procedure in one dimension

The finite difference procedure is illustrate using examples with known solutions. Consider the boundary value problem

$$u_{xx} = f(x)\,, \qquad u(0) = U_0 \quad u(1) = U_1$$

in which $f(x)$ is a prescribed function and U_0 and U_1 are given. The finite difference approximation to the solution of this differential equation requires a uniform dissection of $[0, 1]$ by $(n + 1)$ evenly spaced nodes $0 = x_0 < x_1 < \cdots < x_n = 1$ where $x_k = k/n$. For convenience, let $h = 1/n$ so that $x_k = kh$. In the usual way let u_k denote $u(x_k)$ then

$$u_{xx}(x_k) = \frac{u_{k+1} - 2u_k + u_{k-1}}{h^2} + O(h^2)$$

at each interior point of $[0, 1]$. The boundary conditions are satisfied by asserting that $u_0 = U_0$ and $u_n = U_1$. The finite difference solution for $u(x)$ is now obtained by enforcing the original equation at each interior node of $[0, 1]$, that is,

$$\frac{u_{k+1} - 2u_k + u_{k-1}}{h^2} + O(h^2) = f_k\,, \qquad k = 1, 2\cdots, (n-1)$$

where $f_k = f(x_k)$. When arranged in sequence, these equations become

$$\begin{aligned}
u_2 - 2u_1 + u_0 &= h^2\, f_1 + O(h^4) \\
u_3 - 2u_2 + u_1 &= h^2\, f_2 + O(h^4) \\
u_4 - 2u_3 + u_2 &= h^2\, f_3 + O(h^4) \\
\vdots \quad &= \quad \vdots \\
u_n - 2u_{n-1} + u_{n-2} &= h^2\, f_{n-1} + O(h^4)\,.
\end{aligned}$$

However, u_0 and u_n are known and so the first and last equations are re-expressed in the form

$$\begin{aligned}
u_2 - 2u_1 &= h^2\, f_1 - U_0 + O(h^4) \\
-2u_{n-1} + u_{n-2} &= h^2\, f_{n-1} - U_1 + O(h^4)\,.
\end{aligned}$$

Thus the unknown values u_1, u_2, $\cdots$ u_{n-1} are the solution of the system of linear equations

$$\begin{bmatrix} -2 & 1 & 0 & 0 & \cdots & \cdots & 0 \\ 1 & -2 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & -2 & 1 & 0 & \cdots & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \cdots \\ 0 & \cdots & \cdots & \cdots & 0 & 1 & -2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ \vdots \\ u_{n-1} \end{bmatrix} = \begin{bmatrix} h^2 f_1 - U_0 \\ h^2 f_2 \\ \vdots \\ \vdots \\ h^2 f_{n-1} - U_1 \end{bmatrix} + O(h^4) \tag{2.1}$$

The coefficient matrix of this system has a main diagonal in which each entry is -2 and sub- and super-diagonals in which each entry is 1. All other entries are zero. Matrices of this type are called **tri-diagonal** matrices. The solution of the equation $TX = B$ is now examined when T is a tri-diagonal matrix.

2.1 Tri-diagonal algorithm

Let T be the general tri-diagonal matrix

$$\begin{bmatrix} a_1 & b_1 & 0 & 0 & \cdots & \cdots & 0 \\ c_2 & a_2 & b_2 & 0 & \cdots & \cdots & 0 \\ 0 & c_3 & a_3 & b_3 & 0 & \cdots & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \cdots \\ 0 & \cdots & \cdots & \cdots & 0 & c_{n-1} & a_{n-1} \end{bmatrix}$$

then we describe the steps by which the tri-diagonal system of equations $TX = F$ is solved for X. First, the row operation $R_2 \to R_2 - (c_2/a_1)R_1$ is used to remove c_2 to give the tri-diagonal matrix

$$T^{(1)} = \begin{bmatrix} a_1 & b_1 & 0 & 0 & \cdots & \cdots & 0 \\ 0 & \widehat{a}_2 & b_2 & 0 & \cdots & \cdots & 0 \\ 0 & c_3 & a_3 & b_3 & 0 & \cdots & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \cdots \\ 0 & \cdots & \cdots & \cdots & 0 & c_{n-1} & a_{n-1} \end{bmatrix}$$

where $\widehat{a}_2$ and $\widehat{f}_2$ are defined by

$$a_2 \to a_2 - \frac{c_2}{a_1} b_1 = \widehat{a}_2 \,, \qquad f_2 \to f_2 - \frac{c_2}{a_1} f_1 = \widehat{f}_2 \,.$$

This algorithm is recursive. The second row affects only a_2 and f_2, the second entry of $T^{(1)}$ can now be used to remove c_3 provided $\widehat{a}_2$ is non-zero. This will always be true provided T is non-singular. After $(n-1)$ repetitions of this algorithm, in which the k^{th} repetition operates on the k^{th}

and $(k+1)^{th}$ rows of $T^{(k)}$, the original equations $TX = F$ will be reduced to the new system of equations $T^{(n-1)}X = \widehat{F}$ where

$$T^{(n-1)} = \begin{bmatrix} a_1 & b_1 & 0 & 0 & \cdots & \cdots & 0 \\ 0 & \widehat{a}_2 & b_2 & 0 & \cdots & \cdots & 0 \\ 0 & 0 & \widehat{a}_3 & b_3 & 0 & \cdots & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \cdots \\ 0 & \cdots & \cdots & \cdots & 0 & 0 & \widehat{a}_{n-1} \end{bmatrix}.$$

Since $T^{(n-1)}$ is now an upper triangular matrix, then $T^{(n-1)}X = \widehat{F}$ can be solved for X by backward substitution. the computational effort in this algorithm is directly proportional to N, the number of equations in the system. Another way to think of the tri-diagonal algorithm is that every non-singular tri-diagonal matrix T can be expressed as the product LU where L is a lower triangular matrix and U is an upper triangular matrix. Recall that in general L holds the row operations used in the gaussian elimination algorithm, and that U is the gaussian reduced form of T. Therefore, all the non-zero entries of L are located in its main diagonal and sub-diagonal. Similarly, all the non-zero entries of U are located in its main diagonal and super-diagonal.

2.1.1 Measurement of numerical error

There are two measures of numerical accuracy in common use. To make these measures specific, suppose that the numerical solution and exact solution of $u'' = f(x)$ at the node x_k are respectively $\widehat{u}_k$ and u_k. The first measure, usually called the root-mean-square-error or L^2 error, is defined to be

$$E_n(u - \widehat{u}) = \Big[\frac{1}{n+1} \sum_{k=0}^{n} (u_k - \widehat{u}_k)^2 \Big]^{1/2} \tag{2.2}$$

This summation is approximately proportional to the square root of the integral of the squared difference of the exact and numerical solutions. The second measure, usually called the mean-absolute-error or L^1 error, is defined to be

$$E_n(u - \widehat{u}) = \frac{1}{n+1} \sum_{k=0}^{n} | u_k - \widehat{u}_k | \tag{2.3}$$

This summation is approximately proportional to the integral of the absolute difference between the exact and numerical solutions. Both measures usually paint the same picture, but it is well known that the L^1 error is more discriminating. For convenience we shall always use the L^2 norm for measuring errors and use the notation E_n to denote the value of the L^2 norm when applied to a dissection containing 2^n intervals of uniform length h.

Example Let $u(x)$ be the solution of the boundary value problem

$$u_{xx} = \sin(\alpha x)\,, \qquad u(0) = u(1) = 0$$

in which α is an arbitrary non-zero parameter. This corresponds to the choices $f(x) = \sin(\alpha x)$ and $U_0 = U_1 = 0$. The exact solution to this differential equation is

$$u(x) = \frac{x\sin(\alpha) - \sin(\alpha x)}{\alpha^2}\,.$$

Table 2.1 gives the L^1 and L^2 norms of the numerical solution to this differential equation for various choices of n when $\alpha = 1$ and $\alpha = 50$ respectively.

	$\alpha = 1$		$\alpha = 50$	
n	E_n	E_{n-1}/E_n	E_n	E_{n-1}/E_n
4	0.0002242	–	0.0121297	–
8	0.0000561	3.904	0.0121549	0.960
16	0.0000140	4.001	0.0001235	0.990
32	0.0000035	4.002	0.0000663	98.35
64	0.0000009	4.000	0.0000151	1.860
128	0.0000002	4.000	0.0000037	4.401
256	0.0000001	4.000	0.0000009	4.090
512	0.0000000	4.000	0.0000002	4.020
1024	0.0000000	4.000	0.0000001	4.010

Table 2.1: Comparison of L^2 errors in the solution of $u_{xx} = \sin(\alpha x)$ for the boundary conditions $u(0) = u(1) = 0$ when $\alpha = 1$ and when $\alpha = 50$.

Points to note First, an accurate solution for $\alpha = 1$ and $\alpha = 50$ can be obtained by using a suitably fine mesh. Of course, in practice the finite numerical precision available in the computation will intervene and provide a practical limit to the maximum available accuracy. Second, the ratio of errors has limiting value 4 independent of the value of α. As n is doubled, h is halved, and the error decreases by a factor of 4, that is, the error is proportional to h^2. This is in complete agreement with the presumed accuracy of the finite difference approximation of the second derivative.

2.1.2 Loss of significance - Pivoting

To appreciate how loss of numerical accuracy can occur in the solution of simultaneous equations, consider the use of a computer holding 3 significant figures to solve the equations

$$\begin{aligned} 0.0001\,x_1 \;+\; 1.00\,x_2 &= 1.00 \\ 1.00\,x_1 \;+\; 1.00\,x_2 &= 2.00 \end{aligned}$$

[The actual solution is $x_1 \cong 1.00010$ and $x_2 \cong 0.99990$ to 5 decimal places.]

The matrix formulation of the problem is

$$\begin{bmatrix} 0.0001 & 1.00 \\ 1.00 & 1.00 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1.00 \\ 2.00 \end{bmatrix}$$

Using exact arithmetic the row operation $R_2 \to R_2 - 10000R_1$ yields

$$\begin{bmatrix} 0.0001 & 1.00 \\ 0.0000 & -9999.0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1.00 \\ -9998.0 \end{bmatrix}.$$

However, on a computing machine with 3 significant figures, the second row of these equations cannot be held to the required accuracy and is approximated. The numerical approximation is

$$\begin{bmatrix} 0.0001 & 1.00 \\ 0.0000 & -1.00 \times 10^4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1.00 \\ -1.00 \times 10^4 \end{bmatrix}.$$

The solution of this system is $x_2 = 1.00$ followed by $x_1 = 0.00$. A catastrophic error has occurred due to finite precision arithmetic. The correct procedure is to interchange rows 1 and 2 of the original matrix to obtain

$$\begin{bmatrix} 1.00 & 1.00 \\ 0.0001 & 1.00 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2.00 \\ 1.00 \end{bmatrix}$$

Now perform the row operation $R_2 \to R_2 - 0.0001R_1$ to obtain **in exact arithmetic**

$$\begin{bmatrix} 1.00 & 1.00 \\ 0.00 & 0.9999 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2.00 \\ 0.9998 \end{bmatrix}$$

As previously, on a computing machine with 3 significant figures, the second row of these equations cannot be held to the required accuracy and is approximated. The numerical approximation is

$$\begin{bmatrix} 1.00 & 1.00 \\ 0.00 & 1.00 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2.00 \\ 1.00 \end{bmatrix}$$

which in turn yields the approximate solutions $x_2 = 1.00$ followed by $x_1 = 1.00$. The process of interchanging rows is called **pivoting**. In general, the row in the matrix with the largest entry (in modulus) in the column for which the row is to be used to eliminate other entries is chosen as the pivotal row.

Example Use Gaussian elimination with partial pivoting to solve the equations

$$\begin{aligned} x_1 + 2x_2 - 3x_3 &= 2 \\ x_1 - 2x_2 + x_3 &= 1 \\ -2x_1 + x_2 - x_3 &= 0 \end{aligned}.$$

Solution The augmented matrix is

$$\begin{aligned}
&\begin{bmatrix} 1 & 2 & 3 & 2 \\ 1 & -2 & 1 & 1 \\ -2 & 1 & -1 & 0 \end{bmatrix} \sim \begin{bmatrix} -2 & 1 & -1 & 0 \\ 1 & -2 & 1 & 1 \\ 1 & 2 & 3 & 2 \end{bmatrix} && R_1 \leftrightarrow R_3 \\
&\sim \begin{bmatrix} -2 & 1 & -1 & 0 \\ 0 & -\frac{3}{2} & \frac{1}{2} & 1 \\ 0 & \frac{5}{2} & \frac{5}{2} & 2 \end{bmatrix} && \begin{matrix} R_2 \to R_2 + \frac{1}{2}R_1 \\ R_3 \to R_3 + \frac{1}{2}R_1 \end{matrix} \\
&\sim \begin{bmatrix} -2 & 1 & -1 & 0 \\ 0 & \frac{5}{2} & \frac{5}{2} & 2 \\ 0 & -\frac{3}{2} & \frac{1}{2} & 1 \end{bmatrix} && R_2 \leftrightarrow R_3 \\
&\sim \begin{bmatrix} -2 & 1 & -1 & 0 \\ 0 & \frac{5}{2} & \frac{5}{2} & 2 \\ 0 & 0 & 2 & \frac{11}{5} \end{bmatrix} && R_3 \to R_3 + \frac{3}{5}R_2
\end{aligned}$$

The solution is now obtained by back substitution in the order $x_3 = 1.1$, $x_2 = -0.3$ and $x_1 = -0.7$.

2.2 Equations with non-constant coefficients

The numerical solution of ordinary differential equations with non-constant coefficients follows a similar pattern to the constant coefficient case. The procedure is again illustrated by example. Let $u(x)$ be the solution of the boundary value problem

$$a(x)\,u_{xx} + b(x)\,u_x + c(x)\,u = f(x)\,, \qquad u(x_0) = U_0, \quad u(x_n) = U_1$$

where $a(x), b(x), c(x)$ and $f(x)$ are known functions of x, the constants U_0 and U_1 are given, and finally, x_0 and x_n are the left hand and right hand endpoints of an interval which is uniformly dissected by the points $x_k = x_0 + kh$ where $h = (x_n - x_0)/n$. At the k^{th} internal node of the interval $[x_0, x_n]$, the finite difference representation of the differential equation is

$$a_k\Big[\frac{u_{k+1} - 2u_k + u_{k-1}}{h^2} + O(h^2)\Big] + b_k\Big[\frac{u_{k+1} - u_{k-1}}{2h} + O(h^2)\Big] + c_k u_k = f_k$$

where $a_k = a(x_k)$, $b_k = b(x_k)$, $c_k = c(x_k)$ and $f_k = f(x_k)$. This equation is now multiplied by $2h^2$ and the terms re-arranged to give

$$(2a_k + hb_k)u_{k+1} - (4a_k - 2h^2c_k)u_k + (2a_k - hb_k)u_{k-1} = 2h^2 f_k + O(h^4)$$

for $k = 1, 2, \cdots (n-1)$. As previously, the first and last equations take advantage of the boundary conditions to get the final system of equations

$$\begin{aligned}
-(4a_1 - 2h^2c_1)u_1 + (2a_1 + hb_1)u_2 &= 2h^2 f_1 - (2a_0 - hb_0)U_0 + O(h^4) \\
\vdots &= \vdots \\
(2a_k - hb_k)u_{k-1} - (4a_k - 2h^2c_k)u_k + (2a_k + hb_k)u_{k+1} &= 2h^2 f_k + O(h^4) \\
\vdots &= \vdots \\
(2a_{n-1} - hb_{n-1})u_{n-2} - (4a_{n-1} - 2h^2c_{n-1})u_{n-1} &= 2h^2 f_k - (2a_n + hb_n)U_1 + O(h^4)
\end{aligned} .$$

Example Let $u(x)$ be the solution of the boundary value problem

$$x^2 u_{xx} + x u_x + u = x\,, \qquad u(\varepsilon) = u(1) = 0$$

in which $\varepsilon \in (0,1)$. This equation corresponds to the choices $a(x) = x^2$, $b(x) = f(x) = x$, $c(x) = 1$ and $U_0 = U_1 = 0$. The exact solution to this differential equation is

$$u(x) = \frac{1}{2}\Big[\, x + \frac{\sin(\log x/\varepsilon) - \varepsilon \sin(\log x)}{\sin(\log \varepsilon)} \Big]\,.$$

Table 2.2 gives the L^1 and L^2 norms of the numerical solution to this differential equation for various choices of n when $\varepsilon = 0.01$ and when $\varepsilon = 0.001$.

	$\varepsilon = 0.01$		$\varepsilon = 0.001$	
n	E_n	E_{n-1}/E_n	E_n	E_{n-1}/E_n
4	1.8429306	–	4.4799446	–
8	0.6984934	0.141	0.0312476	0.127
16	0.2161714	2.638	0.3327640	143.4
32	0.0752496	3.231	0.5445338	0.094
64	0.0206367	2.873	0.8809907	0.611
128	0.0045843	3.646	3.9401618	0.618
256	0.0010199	4.502	0.5773118	0.224
512	0.0002431	4.495	0.1126978	6.825
1024	0.0000599	4.195	0.0236637	5.123
4096	0.0000037	4.056	0.0011824	4.762
16384	0.0000002	4.004	0.0000719	4.292
65536	0.0000000	4.000	0.0000045	4.023

Table 2.2: Comparison of L^2 errors in the solution of $x^2u_{xx} + xu_x + u = x$ for the boundary conditions $u(\varepsilon) = u(1) = 0$ when $\varepsilon = 0.01$ and when $\varepsilon = 0.001$.

As previously, it is clear that the finite difference approach based on central difference formulae has error $O(h^2)$ provided h is suitably small. The difficult in this application stems from the fact

that the differential equation has a singular solution at $x = 0$. The closer ε approaches zero, the more troublesome will be the numerical solution. All numerical procedures to solve this differential equation will experience the same difficulty. For example, it is clear that in the vicinity of $x = \varepsilon$ that u_{xx}/u has order ε^{-2} - very large when ε is small. Consequently the error in the finite difference approximation to the derivatives of u around $x = \varepsilon$ will experience a true numerical error of order $(h/\varepsilon)^2$. This is only small when h is very small. The standard way to eliminate problems of this sort is to re-express the differential equation in terms of another independent variable, say $z = \log(x)$ in this particular example.

2.3 Gradient boundary conditions

To date, all boundary conditions have specified the value of the solution on the boundary. Often, however, it is the derivative of the solution that is prescribed on a boundary. Suppose it is required to solve an ODE in $[0, 1]$ with a gradient boundary condition at $x = 1$. We explore several ways to do this.

2.3.1 First order approximation

The first approach is to write

$$u(x+h) = u(x) + h\,u_x(x) + \frac{h^2}{2}u_{xx}(x) + O(h^3)$$

and observe that this equation can be re-expressed in the form

$$u_x(x) = \frac{u(x+h) - u(x)}{h} + O(h)\,.$$

The finite difference representation of a gradient boundary condition at a left or right endpoint is

Left hand boundary	Right hand boundary
$u_x(x_0) = \dfrac{u_1 - u_0}{h} + O(h)$	$u_x(x_n) = \dfrac{u_n - u_{n-1}}{h} + O(h)$

Therefore to use the finite difference algorithm to solve the differential equation

$$a(x)\,u_{xx} + b(x)\,u_x + c(x)\,u = f(x)\,, \qquad u(x_0) = U_0, \quad u_x(x_n) = G_1$$

where $a(x), b(x), c(x)$ and $f(x)$ are known functions of x, and the constants U_0 and G_1 are given, one enforces the ordinary differential equation at each interior node of $[x_0, x_n]$ and uses the finite difference representation of the boundary condition at $x = x_n$. Note that one further equation is now

required because u_n is no longer known and has to be determined as part of the solution process. The finite difference equations are

$$\begin{aligned}
-(4a_1 - 2h^2c_1)u_1 + (2a_1 + hb_1)u_2 &= 2h^2f_1 - (2a_1 - hb_1)U_0 + O(h^4) \\
\vdots &= \vdots \\
(2a_k - hb_k)u_{k-1} - (4a_k - 2h^2c_k)u_k + (2a_k + hb_k)u_{k+1} &= 2h^2f_k + O(h^4) \\
\vdots &= \vdots \\
(2a_{n-1} - hb_{n-1})u_{n-2} - (4a_{n-1} - 2h^2c_{n-1})u_{n-1} + (2a_n + hb_n)u_n &= 2h^2f_k + O(h^4) \\
-u_{n-1} + u_n &= hG_1 + O(h^2)
\end{aligned}$$

To appreciate how the inferior accuracy of the last equation degrades the accuracy of the entire solution, it is sufficient to find the numerical solution of

$$x^2u_{xx} + xu_x + u = x\,, \qquad u(\varepsilon) = u'(1) = 0$$

with $\varepsilon = 0.01$. This equation corresponds to the choices $a(x) = x^2$, $b(x) = f(x) = x$, $c(x) = 1$ and $U_0 = G_1 = 0$ and has exact solution

$$u(x) = \frac{1}{2}\left[\, x + \frac{\sin(\log\varepsilon/x) - \varepsilon\cos(\log x)}{\cos(\log\varepsilon)} \,\right].$$

Table 2.3 illustrates the error structure of the numerical solution based on the finite difference method.

n	E_n	E_{n-1}/E_n
4	3.7369443	–
8	3.4181676	1.246
16	3.1350210	1.093
32	2.5472682	1.090
64	1.4788262	1.231
128	0.5404178	1.722
256	0.1713286	2.736
512	0.0598533	3.154
1024	0.0235584	2.862
2048	0.0101889	2.541
4096	0.0046959	2.312
8192	0.0022481	2.169
32768	0.0005433	2.089

Table 2.3: Comparison of L^2 errors in the solution of $x^2u_{xx}+xu_x+u = x$ where $u(0.01) = u'(1) = 0$. The gradient boundary condition is implemented to $O(h)$ accuracy.

It is clear from Table 2.3 that the accuracy of the finite difference scheme is degraded to the accuracy of its worst equation - in this case the boundary condition equation which is $O(h)$ accurate. We need $O(h^2)$ accurate forward and backward difference formulae for gradients.

2.3.2 Second order approximation

The $O(h^2)$ accurate forward and backward finite difference formulae for $u_x(x)$ are respectively

$$\begin{aligned} -3u(x) + 4u(x+h) - u(x+2h) &= 2hu_x(x) + O(h^3)\ , \\ u(x-2h) - 4u(x-h) + 3u(x) &= 2hu_x(x) + O(h^3)\ . \end{aligned}$$

Thus the $O(h^2)$ accurate finite difference representation of a gradient at a left or a right endpoint is

Left hand boundary	Right hand boundary
$u_x(x_0) = \dfrac{-3u_0 + 4u_1 - u_2}{2h} + O(h^2)$	$u_x(x_n) = \dfrac{u_{n-2} - 4u_{n-1} + 3u_n}{2h} + O(h^2)$

The new finite difference formulation of the original problem only differs from the original formulation in respect of one equation, namely that contributed by the boundary condition at $x = 1$. The numerical form is

$$\begin{aligned} -(4a_1 - 2h^2c_1)u_1 + (2a_1 + hb_1)u_2 &= 2h^2f_1 - (2a_0 - hb_0)U_0 + O(h^4) \\ \vdots &= \vdots \\ (2a_k - hb_k)u_{k-1} - (4a_k - 2h^2c_k)u_k + (2a_k + hb_k)u_{k+1} &= 2h^2f_k + O(h^4) \\ \vdots &= \vdots \\ (2a_{n-1} - hb_{n-1})u_{n-2} - (4a_{n-1} - 2h^2c_{n-1})u_{n-1} + (2a_n + hb_n)u_n &= 2h^2f_k + O(h^4) \\ u_{n-2} - 4u_{n-1} + 3u_n &= 2hG_1 + O(h^3) \end{aligned}$$

Unfortunately, the matrix of the resulting system of equations is no longer tri-diagonal, but this causes no difficulty since the matrix can be converted to tri-diagonal form by a row operation in which the penultimate row of the matrix is used to eliminate u_{n-2} from the last row of the matrix.

n	E_n	E_{n-1}/E_n	n	E_n	E_{n-1}/E_n
4	3.7567566	–	512	0.0261418	4.060
8	3.4135978	1.116	1024	0.0064812	4.097
16	3.1071541	1.101	2048	0.0016165	4.033
32	2.4869724	1.099	4096	0.0004039	4.009
64	1.3708621	1.249	8192	0.0001009	4.002
128	0.4348676	1.814	16384	0.0000252	4.001
256	0.1071061	3.152	32768	0.0000063	4.000

Table 2.4: L^2 errors in the solution of $x^2u_{xx} + xu_x + u = x$ with boundary conditions $u(0.01) = u_x(1) = 0$ are compared. The gradient boundary condition is implemented to $O(h^2)$ accuracy.

Table 2.4 illustrates the behaviour of the numerical solution based on the finite difference method in which the gradient boundary condition is implemented to $O(h^2)$ accuracy. Clearly $O(h^2)$ accuracy is now recovered by using the second order accurate implementation of the boundary condition.

2.3.3 Fictitious nodes

The most direct way to derive a second order accurate boundary condition is to use the central difference formulae

$$u_x(x_0) = \frac{u_1 - u_{-1}}{2h} + O(h^2)\,, \qquad u_x(x_n) = \frac{u_{n+1} - u_{n-1}}{2h} + O(h^2)\,.$$

The difficulty with these approximations is that they introduce fictitious solutions u_{-1} and u_{n+1} at the fictitious nodes x_{-1} (for $u_x(x_0)$) and x_{n+1} (for $u_x(x_n)$) respectively. Both nodes are fictitious because they lie **outside** the region in which the differential equation is valid. The introduction of a fictitious node creates another unknown to be determined. The procedure is as follows. The differential equation is assumed to be valid at the node x_n to obtain

$$a_n\Big[\,\frac{u_{n+1} - 2u_n + u_{n-1}}{h^2} + O(h^2)\,\Big] + b_n G_1 + c_n u_n = f_n\,, \qquad \frac{u_{n+1} - u_{n-1}}{2h} = G_1 + O(h^2)\,.$$

The task is now to eliminate u_{n+1} between both equations. To do this, multiply the first equation by h^2 and the second equation by $2ha_n$ and subtract to get

$$2a_n u_{n-1} + (h^2 c_n - 2a_n)u_n = h^2 f_n - (2a_n + hb_n)hG_1 + O(h^3)\,.$$

This equation is now used as the final equation of the tri-diagonal system which becomes

$$\begin{aligned}
-(4a_1 - 2h^2c_1)u_1 + (2a_1 + hb_1)u_2 &= 2h^2 f_1 - (2a_0 - hb_0)U_0 + O(h^4)\\
\vdots &= \qquad\qquad \vdots\\
(2a_k - hb_k)u_{k-1} - (4a_k - 2h^2c_k)u_k + (2a_k + hb_k)u_{k+1} &= 2h^2 f_k + O(h^4)\\
\vdots &= \qquad\qquad \vdots\\
(2a_{n-1} - hb_{n-1})u_{n-2} - (4a_{n-1} - 2h^2c_{n-1})u_{n-1} + (2a_n + hb_n)u_n &= 2h^2 f_k + O(h^4)\\
2a_n u_{n-1} + (h^2 c_n - 2a_n)u_n &= h^2 f_n - (2a_n + hb_n)hG_1 + O(h^3)
\end{aligned}$$

Table 2.5 compares the error structure of the backward difference and fictitious node implementations of the gradient boundary condition when $\varepsilon = 0.01$.

	Backward difference		Fictitious nodes	
n	E_n	E_{n-1}/E_n	E_n	E_{n-1}/E_n
4	3.7567566	–	3.7573398	–
8	3.4135978	1.116	3.4144871	1.234
16	3.1071541	1.101	3.1108516	1.100
32	2.4869724	1.099	2.4917194	1.098
64	1.3708621	1.249	1.3755215	1.248
128	0.4348676	1.814	0.4371938	1.811
256	0.1071061	3.152	0.1078104	3.146
512	0.0261418	4.060	0.0263256	4.055
1024	0.0064812	4.097	0.0065275	4.095
2048	0.0016165	4.033	0.0016281	4.033
4096	0.0004039	4.009	0.0004068	4.009
8192	0.0001009	4.002	0.0001017	4.002
16384	0.0000252	4.001	0.0000254	4.001
32768	0.0000063	4.000	0.0000064	4.000

Table 2.5: Comparison of L^2 errors in the solution of $x^2u'' + xu' + u = x$ for the boundary conditions $u(0.01) = u'(1) = 0$. The results on the left use the backward difference implementation and those on the right use the fictitious nodes implementation of the gradient boundary condition $u'(1) = 0$. Both implementations are $O(h^2)$ accuracy.

2.3.4 Richardson extrapolation

To date we have assumed that n is given, although in reality the value of n is unknown and one does not have an exact solution with which to assess the accuracy of the procedure. We illustrate the idea of Richardson extrapolation. The success of the method lies in precise knowledge of the error structure of the numerical procedure - in this case the procedure has error $O(h^2)$.

Example Construct the finite difference equations for the numerical solution of the boundary value problem

$$u'' - u' \tan x = -1\,, \qquad u(0) = 0\,, \quad u'(\pi/4) = -1\,.$$

The exact solution is $u(x) = \log\cos x$.

Solution Take $h = \pi/(4n)$ then $a(x) = 1$, $b(x) = -\tan x$, $c(x) = 0$ and $f(x) = -1$. The boundary conditions are $U_0 = 0$ and $G_1 = -1$. The finite difference equations based on the fictitious nodes

procedure are therefore

$$\begin{aligned}
-4u_1 + (2+hb_1)u_2 &= 2h^2 f_1 + O(h^4) \\
\vdots &= \vdots \\
(2-hb_k)u_{k-1} - 4u_k + (2+hb_k)u_{k+1} &= 2h^2 f_k + O(h^4) \\
\vdots &= \vdots \\
(2-hb_{n-1})u_{n-2} - 4u_{n-1} + (2+hb_n)u_n &= 2h^2 f_k + O(h^4) \\
2u_{n-1} - 2u_n &= h^2 f_n + (2+hb_n)h + O(h^3)
\end{aligned} \quad .$$

Suppose initially that $n = 40$ - the initial choice of n is arbitrary but should be sensible. Let the finite difference solution computed at these nodes be $u_0^{(1)}, \cdots, u_n^{(1)}$. Now take $n = 2n$ and recompute the new solution. Let this solution be $u_0^{(2)}, \cdots, u_{2n}^{(2)}$. The solution $u_k^{(1)}$ may be compared directly with the solution $u_{2k}^{(2)}$ since both apply at the same node. The root mean square

$$E_n = \Big[\frac{1}{n+1} \sum_{k=0}^{n} (u_k^{(1)} - u_{2k}^{(2)})^2 \Big]^{1/2}$$

is a measure of the error incurred in computing the original solution. In this case E_{40} is determined. The procedure may be continued by doubling n again - E_{80} is now computed. The algorithm continues to double n and terminates whenever $E_n < \varepsilon$ where ε is a user supplied error bound. Moreover, since we already know that the procedure is $O(h^2)$, we expect the ratio E_{n-1}/E_n to have a limiting value of 4 so that every iteration of this loop approximately reduces E_n by a factor of 4. Table 2.6 illustrates the convergence properties of E_n.

n	E_n	E_{n-1}/E_n
40	0.0001026	–
80	0.0000261	3.922
160	0.0000066	3.960
320	0.0000017	3.980
640	0.0000004	3.990

Table 2.6: Estimated L^2 norm of the errors incurred in the solution of $u_{xx} - \tan x u_x = -1$ with boundary conditions $u(0) = 0$ and $u_x(\pi/4) = -1$.

Richardson extrapolation takes advantage of the error structure of the solution. Suppose $u^{(1)}$ and $u^{(2)}$ are two approximations of the exact solution u corresponding to dissection refinements h_1 and h_2 respectively. Since the error is $O(h^2)$ then

$$\left.\begin{aligned} u^{(1)} &= u + ch_1^2 + O(h_1^3) \\ u^{(2)} &= u + ch_2^2 + O(h_2^3) \end{aligned}\right] \quad \rightarrow u = \frac{h_2^2\, u^{(1)} - h_1^2\, u^{(2)} + O(h_2^2 h_1^3 - h_1^2 h_2^3)}{h_2^2 - h_1^2} \, .$$

The error in the combination of $u^{(1)}$ and $u^{(2)}$ defined by u is more accurate than either of $u^{(1)}$ or

$u^{(2)}$. In the case in which $h_2 = h_1/2$, it is easy to see that the extrapolated solution is

$$u = \frac{4\,u^{(2)} - u^{(1)}}{3} + O(h^3)\,.$$

To test this idea, the Richardson extrapolation is used to improve the estimated solution and the L^2 norm is computed with respect to the exact solution so as to check the improved convergence. To be specific, $u^{(1)}$ and $u^{(2)}$ are combined to give the more accurate solution $\widehat{u}^{(1)}$ which is then used for the norm computation. In the next stage, $u^{(2)}$ and $u^{(3)}$ are used to give the more accurate solution $\widehat{u}^{(2)}$ which is then used for the norm computation. The procedure is repeated. Table 2.7 records the results of these calculations.

n	E_n	E_{n-1}/E_n
40	0.1166×10^{-5}	–
80	0.1493×10^{-6}	7.810
160	0.1890×10^{-7}	7.902
320	0.2378×10^{-8}	7.946
640	0.3029×10^{-9}	7.851

Table 2.7:
Estimates the L^2 norm of the errors incurred in the solution of $u_{xx} - \tan x u_x = -1$ with boundary conditions $u(0) = 0$ and $u_x(\pi/4) = -1$. The norm is computed from the exact solution and the Richardson extrapolation of the finite difference solution.

2.4 Non-linear equations

The application of the finite difference algorithm to non-linear equations is similar to that of linear equations in the respect that the non-linear equation is enforced at each node of the dissection by replacing each derivative appearing in the equation with its finite difference representation. However, the resulting equations are no longer linear. The general solution strategy is based on iteration. The algebraic equations resulting from the replacement of derivatives by finite differences are expressed in a fixed-point format that is known to converge.

2.4.1 Fixed point procedure

One dimension

The general idea of a fixed point procedure is first illustrated for the solution of $f(x) = 0$ assuming the equation has a solution near x_0. The aim is to construct a fixed point scheme which converges to the true solution, assumed to be near x_0. Let λ be an arbitrary constant then the identity

$$x = x + \lambda f(x) = g(x)$$

may be used to construct the iterative scheme $x_{k+1} = g(x_k)$ with initial value x_0. If this scheme converges, then the limit point is a solution of $f(x) = 0$. The usual way to promote convergence is to choose λ so that $g(x)$ is a contraction mapping near the fixed point. If x_0 is a good approximation to this fixed point, then choose λ so that $g'(x_0) = 1 + \lambda f'(x_0) = 0$, that is, $\lambda = -1/f'(x_0)$. The iteration is now

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_0)}\,, \qquad x_0 \text{ given.}$$

Note that this algorithm is similar to the Newton-Raphson algorithm, the latter simply updating λ as improved estimates of the fixed point are found.

Many dimensions

Let $\mathbf{F}(\mathbf{x}) = 0$ be a system of n equations in n unknowns, and suppose that X_0 is an approximation to this solution. The generalisation of the one dimensional algorithm to many dimensions begins with the identity

$$X = X + AF(X) = G(X)$$

in which X is a vector of dimension n and A is an arbitrary constant $n \times n$ matrix. This identity in turn leads to the iterative algorithm $X_{k+1} = X_k + AF(X_k)$. To make this a contraction mapping in the vicinity of X_0, it is enough to choose A such that

$$\frac{\partial(X + AF(X))}{\partial X_j}\Big|_{X=X_0} = 0 \quad \rightarrow \quad I + AJ(X_0) = 0$$

where $J(X_0)$ is the Jacobian of $F(X)$ evaluated at X_0. Clearly $A = -J^{-1}(X_0)$ and the final iterative scheme is

$$X_{k+1} = X_k - J^{-1}(X_0)F(X_k)\,, \qquad X_0 \text{ given}\,.$$

Provided X_0 is a good guess of the original solution then this algorithm should converge to the exact solution. The Newton-Raphson variant of this scheme is obtained by upgrading A whenever each new estimate of the zero becomes available. Of course, $J^{-1}(X_0)$ is never computed in this algorithm; instead we solve the simultaneous equations

$$J(X_0)(X_{k+1} - X_k) = -F(X_k) \qquad X_0 \text{ given}\,.$$

Example Find the solutions of the simultaneous equations

$$xy - \sin x - y^2 = 0\,, \qquad x^2y^3 - \tan y - 1 = 0\,.$$

using the Newton-Raphson algorithm.

Solution Here $F(X) = [xy - \sin x - y^2, x^2y^3 - \tan y - 1]$ and the Jacobian of this vector is

$$J(X) = \begin{bmatrix} y - \cos x & x - 2y \\ 2xy^3 & 3x^2y^2 - \sec^2 y \end{bmatrix}$$

With starting values $X = [0.0, -1.0]$ the first 10 iterations assuming a fixed A and with A updated as iteration proceeds (Newton-Raphson) are given in Table 2.8.

	A not updated		Newton-Raphson	
n	x	y	x	y
1	−0.3372779	−0.8372779	−0.3372779	−0.8372779
2	−0.3686518	−0.8247921	−0.3762218	−0.8235464
3	−0.3748875	−0.8230965	−0.3767514	−0.8235034
4	−0.3762579	−0.8230912	−0.3767515	−0.8235034
⋮	⋮	⋮	⋮	⋮
15	−0.3767514	−0.8235032	−0.3767515	−0.8235034
16	−0.3767515	−0.8235034	−0.3767515	−0.8235034

Table 2.8: Estimated solutions of the simultaneous equations $xy - \sin x - y^2 = 0$ and $x^2y^3 - \tan y - 1 = 0$ using a fixed A and a continually updated A (Newton-Raphson).

Example Use the finite difference algorithm to solve the boundary value problem

$$u_{xx} = u^2\,, \qquad u(0) = 0, \quad u(1) = 1\,.$$

Solution In the usual notation, the finite difference approximation to this equation is

$$u_{k+1} - 2u_k + u_{k-1} - h^2u_k^2 = 0\,, \qquad k = 1, \cdots, (n-1)$$

where $u_0 = 0$ and $u_n = 1$. These values appear in the first and last equations and so F is

$$\begin{aligned} F_1 &= u_2 - 2u_1 - h^2u_1^2 \\ \vdots &= \vdots \\ F_k &= u_{k+1} - 2u_k + u_{k-1} - h^2u_k^2 \\ \vdots &= \vdots \\ F_{n-1} &= 1 - 2u_{n-1} + u_{n-2} - h^2u_{n-1}^2 \end{aligned} \quad .$$

The Jacobian matrix is $J_{ik} = F_{i,k}$. In this example, it is clear that J is tri-diagonal. To be specific,

$$J = \begin{bmatrix} -2-2h^2u_1 & 1 & 0 & 0 & \cdots & \cdots & 0 \\ 1 & -2-2h^2u_2 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & -2-2h^2u_3 & 1 & 0 & \cdots & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \cdots \\ \vdots & \vdots & \ddots & 1 & -2-2h^2u_k & 1 & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \cdots \\ 0 & \cdots & \cdots & \cdots & 0 & 1 & -2-2h^2u_{n-1} \end{bmatrix}$$

The algorithm is started by guessing the initial solution. Often a suitable guess is obtained by assuming any sensible function of x connecting the boundary data - say $u(x) = x$ in this instance. The refinement of the mesh is set by choosing N and the initial estimate of $u_0, u_1, \cdots, u_n$ determined from the guessed solution. The value of n is then doubled and the computation repeated. The root mean square error norm

$$E_n = \Big[\frac{1}{n+1} \sum_{k=0}^{n} (u_k^{(1)} - u_{2k}^{(2)})^2 \Big]$$

is then used to decide when the algorithm should terminate. With a stopping criterion of $E_n < 10^{-8}$, Table 2.9 shows the result of twelve iterations of the Newton-Raphson algorithm.

N	E_N	E_{N-1}/E_N
4	0.0001037	–
8	0.0000262	3.959
16	0.0000066	3.992
32	0.0000016	3.998
64	0.0000004	4.000

Table 2.9:
Estimated solutions of the $u_{xx} = u^2$ with boundary conditions $u(0) = 0$ and $u(1) = 1$. The accuracy is based on 12 iterations of the Newton-Raphson algorithm.

Exercises

Question 11. Use the $O(h^2)$ method of finite differences to solve the boundary value problem

$$y'' + xy = 1, \quad y(0) = 0, \quad y(1) = 1$$

taking $h = 1/4$. This means that we want to know the values of $y(0)$, $y(1/4)$, $y(1/2)$, $y(3/4)$ and $y(1)$. Some are given and some you must find.

Question 12. Use the $O(h^2)$ method of finite differences to solve the boundary value problem

$$y'' + y = 0, \qquad y(0) = 1, \quad y'(1) = 0$$

taking $h = 1/4$. [Use the backward difference representation of a first derivative to deal with the gradient boundary condition.]

Show that $y(x) = \sec 1 \cos(1 - x)$ is the exact solution of this equation and use it to check the accuracy of your answers.

Question 13. Use the $O(h^2)$ method of finite differences with $h = 1/3$ to solve the boundary value problem

$$x^2y'' - xy' + y = x^2, \qquad y(1) = 0, \quad y(2) = 4\,.$$

Verify that the exact solution to this boundary value problem is

$$y(x) = x^2 - x + \frac{x \log x}{\log 2}$$

and use this exact solution to check the accuracy of your numerical calculations.

Question 14. Use the $O(h^2)$ method of finite differences with $h = 1/3$ to solve the boundary value problem

$$x^2y'' - xy' + y = x^2, \qquad y'(1) = 0, \quad y(2) = 1\,.$$

Use both forward differences and fictitious nodes to deal with $y'(1) = 0$. Verify that the exact solution to this boundary value problem is

$$y(x) = x^2 + \frac{x(\log x + 3 - 4\log 2)}{2(\log 2 - 1)}$$

and use this exact solution to check the accuracy of your numerical calculations.

Question 15. Show that the finite difference algorithm

$$\begin{aligned} -2u_1 + u_2 &= -U_0 + \frac{h^2}{12}\Big[\, f_0 + 10f_1 + f_2 \,\Big] \\ &\vdots \\ u_{k-1} - 2u_k + u_{k+1} &= \frac{h^2}{12}\Big[\, f_{k-1} + 10f_k + f_{k+1} \,\Big] \\ &\vdots \\ u_{n-2} - 2u_{n-1} &= -U_1 + \frac{h^2}{12}\Big[\, f_{n-2} + 10f_{n-1} + f_n \,\Big] \end{aligned}$$

for the solution of the differential equation $u_{xx} = f(x)$ with boundary conditions $u(0) = U_0$ and $u(1) = U_1$ is $O(h^4)$ accurate.

Question 16. What is the leading error term in the finite difference formula

$$\frac{\partial^2 u(x,y)}{\partial x \partial y} \simeq \frac{u(x+h,y+h) - u(x+h,y-h) - u(x-h,y+h) + u(x-h,y-h)}{4h^2}?$$

[Think about it before expanding, also see degree exam 2000!]

Question 17. Consider the differential equation $x^2 u_{xx} + x u_x + u = x$ with boundary conditions $u(\epsilon) = u(1) = 0$. Use the substitution $x = e^t$ to reformulate the problem as one for $u(t)$. Now discretise in t (using second order central differences). Using $N = 2, 4, ...$ (just a few by hand/calculator or quite a few using a PC) compare results with those given in the lecture. Why is this better?

Question 18. Write down the central difference/Newton Raphson method for the non-linear equation

$$u_{xx} + 2uu_x + u = \sin 2x, \quad u(0) = u(\pi) = 0.$$

[Can you spot one exact solution?]

Chapter 3

Elliptic equations in two dimensions

The finite difference algorithm can be applied to the solution of partial differential equations. The procedure will be demonstrated with respect to partial differential equations in two dimensions. Let $\mathcal{D} \subset \mathbb{R}^2$ be such that $f = f(x, y)$ has Taylor's series expansion

$$\begin{aligned} f(x+h, y+k) &= f(x,y) + hf_x + kf_y + \frac{1}{2}\Big[h^2 f_{xx} + 2hkf_{xy} + k^2 f_{yy}\Big] \\ &\quad + \frac{1}{6}\Big[h^3 f_{xxx} + 3h^2 k f_{xxy} + 3hk^2 f_{xyy} + k^3 f_{yyy}\Big] + \cdots \end{aligned}$$

at $(x, y) \in \mathcal{D}$ for suitable values of h and k. Suppose that $\mathcal{D}$ can be divided into rectangles by the nodes $(x_0 + ih, y_0 + jk)$ where $i = 0, 1, \cdots n$ and $j = 0, 1, \cdots, m$. In particular, the usual central difference approximations are valid and give

$$\begin{aligned} \frac{\partial f(x_i, y_j)}{\partial x} &= \frac{f_{i+1,j} - f_{i-1,j}}{2h} + O(h^2) \\ \frac{\partial f(x_i, y_j)}{\partial y} &= \frac{f_{i,j+1} - f_{i,j-1}}{2k} + O(k^2) \\ \frac{\partial^2 f(x_i, y_j)}{\partial x^2} &= \frac{f_{i+1,j} - 2f_{i,j} + f_{i-1,j}}{h^2} + O(h^2) \\ \frac{\partial^2 f(x_i, y_j)}{\partial y^2} &= \frac{f_{i,j+1} - 2f_{i,j} + f_{i,j-1}}{k^2} + O(k^2) \end{aligned}$$

where $f_{i,j} = f(x_0 + ih, y_0 + jk)$. The only new feature to arise in two dimensions is the mixed derivative f_{xy}. It may be shown that

$$\frac{\partial^2 f(x_i, y_j)}{\partial x \partial y} = \frac{f_{i+1,j+1} - f_{i-1,j+1} - f_{i+1,j-1} + f_{i-1,j-1}}{4hk} + O(hk)\,.$$

To maintain simplicity, it will henceforth be assumed that $h = k$, that is, the resolutions of the mesh in the x and y directions are identical so that the region in which the solution is sought is now partitioned into squares of side h by the nodes. The simplified finite difference formulae for the first

and second order partial derivatives of f become

$$\begin{aligned}
\frac{\partial f(x_i,y_j)}{\partial x} &= \frac{f_{i+1,j}-f_{i-1,j}}{2h}+O(h^2)\\
\frac{\partial f(x_i,y_j)}{\partial y} &= \frac{f_{i,j+1}-f_{i,j-1}}{2h}+O(h^2)\\
\frac{\partial^2 f(x_i,y_j)}{\partial x^2} &= \frac{f_{i+1,j}-2f_{i,j}+f_{i-1,j}}{h^2}+O(h^2)\\
\frac{\partial^2 f(x_i,y_j)}{\partial y^2} &= \frac{f_{i,j+1}-2f_{i,j}+f_{i,j-1}}{h^2}+O(h^2)\\
\frac{\partial^2 f(x_i,y_j)}{\partial x\partial y} &= \frac{f_{i+1,j+1}-f_{i-1,j+1}-f_{i+1,j-1}+f_{i-1,j-1}}{4h^2}+O(h^2)\,.
\end{aligned}$$

3.1 Computational molecule

The computational molecule is a convenient way to describe the complex mix of variables necessary to construct the finite difference representation of a differential operator. For example, Laplace's equation at (x_i, y_j) in two dimensions has finite difference form

$$\begin{aligned}
u_{xx}+u_{yy} &= \frac{u_{i+1,j}-2u_{i,j}+u_{i-1,j}}{h^2}+\frac{u_{i,j+1}-2u_{i,j}+u_{i,j-1}}{h^2}+O(h^2)\\
&= \frac{u_{i+1,j}+u_{i,j+1}-4u_{i,j}+u_{i,j-1}+u_{i-1,j}}{h^2}+O(h^2)=0\,.
\end{aligned}$$

The centre of the molecule is positioned at the node at which the operator is to be expressed and the entries of the molecule indicate the proportion of the surrounding nodes to be mixed in order to obtain the finite difference representation of the operator. Assuming the molecular orientation

$(i-1,j+1)$	$(i,j+1)$	$(i+1,j+1)$
$(i-1,j)$	(i,j)	$(i+1,j)$
$(i-1,j-1)$	$(i-1,j-1)$	$(i-1,j-1)$

the Laplace operator corresponds to the computational molecule

$$u_{xx}+u_{yy}=\frac{1}{h^2}\begin{array}{|ccc|}\hline 0 & 1 & 0\\ 1 & -4 & 1\\ 0 & 1 & 0\\ \hline\end{array}$$

Similarly, the mixed derivative operator corresponds to the computational molecule

$$u_{xy}=\frac{1}{4h^2}\begin{array}{|ccc|}\hline -1 & 0 & 1\\ 0 & 0 & 0\\ 1 & 0 & -1\\ \hline\end{array}$$

Example Construct the finite difference scheme to solve the Poisson equation

$$u_{xx} + u_{yy} = f(x, y)$$

in the domain $\mathcal{D} = [(x, y) \in (0, 1) \times (0, 1)]$ subject to the boundary condition $u = 0$ on $\partial\mathcal{D}$.

Solution The computational molecule at each interior node gives the set of equations

$$u_{i+1,j} + u_{i,j+1} - 4u_{i,j} + u_{i,j-1} + u_{i-1,j} = h^2 f_{i,j}$$

where $(i, j) \in (1, \cdots, n-1) \times (1, \cdots, n-1)$. One must solve a set of $(n-1)^2$ linear equations for the solution at each internal node of $\mathcal{D}$. The fundamental difference between the one-dimensional and higher dimensional finite difference schemes lies in the fact that any node has more than two neighbours contributing to the numerical solution (in fact 4 in the case of the Poisson equation). Consequently, the tri-diagonal structure of the one-dimensional problem is lost since nodes close geographically to each other can no longer be numbered sequentially. An enumeration scheme for the unknowns $u_{i,j}$ is required. The choice is ultimately arbitrary, but one obvious possibility is

$$v_{i+(n-1)*(j-1)} = u_{i,j}\,, \qquad (i, j) \in (1, \cdots, n-1) \times (1, \cdots, n-1)\,.$$

The vector V of unknowns has dimension $(n-1)^2$ and enumerates the nodes from left to right (in the direction of increasing x) and from bottom to top (in the direction of increasing y). The vector V is the solution of the linear system $AV = F$ where A is a constant matrix and F is a vector of dimension $(n-1)^2$ with k^{th} entry $F_k = h^2 f_{i,j}$. The k^{th} row of A will contain the elements of the Laplacian computational module appropriately positioned. The different possibilities are

(a) If $i = 1$ then $A_{k,k} = -4$ and $A_{k,k+1} = 1$.

(b) If $1 < i < (n-1)$ then $A_{k,k-1} = 1$, $A_{k,k} = -4$ and $A_{k,k+1} = 1$.

(c) If $i = n-1$ then $A_{k,k-1} = 1$ and $A_{k,k} = -4$.

For each value of k, the value of j is examined. If $j = 1$ then $A_{k,k+n-1} = 1$, while if $1 < j < (n-1)$ then $A_{k,k-n+1} = A_{k,k+n-1} = 1$, and finally if $j = n-1$ then $A_{k,k-n+1} = 1$. The structure of A may be represented efficiently by the block matrix form

$$A = \begin{bmatrix} T & I & 0 & 0 & \cdots & \cdots & 0 \\ I & T & I & 0 & \cdots & \cdots & 0 \\ 0 & I & T & I & 0 & \cdots & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \cdots \\ 0 & \cdots & \cdots & \cdots & 0 & I & T \end{bmatrix}$$

where I is the $(n-1)\times(n-1)$ identity matrix and T is the $(n-1)\times(n-1)$ tri-diagonal matrix

$$T = \begin{bmatrix} -4 & 1 & 0 & 0 & \cdots & \cdots & 0 \\ 1 & -4 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & -4 & 1 & 0 & \cdots & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \cdots \\ 0 & \cdots & \cdots & \cdots & 0 & 1 & -4 \end{bmatrix}.$$

Suppose now that Gaussian elimination is used to solve these equations. It is clear that the process generates non-zero entries where none entry existed previously. Thus Gaussian elimination is no longer appropriate and another approach, one that can take advantage of the sparsity of A, is required.

3.2 Iterative methods

Iterative methods are most appropriate for the solution of equations $AX = B$ when A is a sparse matrix. A matrix A is **sparse** whenever it is beneficial to store only the non-zero elements of A. Various procedures are used to store sparse matrices; some are general and some are specific to particular matrix types (*e.g.* tri-diagonal matrices).

3.2.1 A general representation of a sparse matrix

The non-zero elements of a general sparse matrix are often stored as a vector together with the row and column indices in which they appear. However, the way in which this is achieved involves a degree of subtlety. The entries of A are stored sequentially row-by-row together with the associated column index. An auxiliary vector, say $\mathbf{r}$, of dimension one more than that of the number of rows of A stores the number of non-zero elements in the k^{th} row of A indirectly in the difference $r_{k+1} - r_k$. This representation allows efficient multiplication by the matrix A.

Example Find the sparse representation of the matrix

$$A = \begin{bmatrix} 0.0 & 1.0 & 2.3 & 0.0 & 0.0 \\ 2.0 & 0.0 & 0.3 & 3.1 & 0.0 \\ 0.0 & -1.0 & -0.4 & 0.0 & 1.5 \\ 0.0 & 0.0 & 0.0 & 3.4 & 0.0 \\ 0.2 & 0.0 & 2.0 & 0.0 & 0.0 \end{bmatrix}$$

Solution The representation consists of the vectors

$$\begin{aligned} \mathbf{a} &= [1.0, 2.3, 2.0, 0.3, 3.1, -1.0, -0.4, 1.5, 3.4, 0.2, 2.0] \\ \mathbf{c} &= [2, 3, 1, 3, 4, 2, 3, 5, 4, 1, 3] \\ \mathbf{r} &= [0, 2, 5, 8, 9, 11] \end{aligned}$$

In practice, these would be enclosed in a structure.

3.2.2 The algorithms

General iterative methods for the solution of $AX = B$ usually begin by rewriting this equation in the iterative form

$$PX + (A - P)X = B$$

where P is an arbitrary non-singular matrix of the same dimension as A. Thereafter the solution of $AX = B$ is derived by iterating

$$PX_{k+1} = B - (A - P)X_k\,, \qquad X_0 \text{ given}\,. \tag{3.1}$$

The efficacy of this iterative algorithm requires that P possess two important properties of P.

- Equation (3.1) must be easy to solve for X_{k+1}. For example, this is true if P is a diagonal (Gauss-Jacobi), an upper triangular matrix (solution by backward substitution) or a lower triangular matrix (Gauss-Seidel) for which the solution is obtained by forward substitution.
- The matrix $I - P^{-1}A$ must be a contraction mapping. This requires the eigenvalues of $I - P^{-1}A$ to lie within the unit circle. Clearly a particularly good choice for P occurs when P^{-1} is a good estimate for the inverse of A.

Let $A = D + L + U$ where D is the main diagonal of A, and L and U denote respectively the lower and upper triangular sections of A. Two important choices for P are:-

Algorithm	P	$H = I - P^{-1}A$
Gauss-Jacobi	D	$-D^{-1}(L + U)$
Gauss-Seidel	$D + L$	$-(D + L)^{-1}U$

The iterative scheme (3.1) will converge to the solution of $AX = B$ from any starting value X_0 provided $|| I - P^{-1}A || < 1$, that is, $(I - P^{-1}A)$ is a contraction mapping in the vector space of the solution. This requires that all the eigenvalues of $(I - P^{-1}A)$ lie within the unit circle. The proof of this result relies on the fact that every matrix is unitarily similar to an upper triangular matrix.

Gauss-Jacobi algorithm The Gauss-Jacobi algorithm corresponds to the choice $P = D$. With this choice of P, it follows that $H = (I - P^{-1}A) = -D^{-1}(L + U)$ and the iterative scheme is

$$X_{n+1} = D^{-1}B - D^{-1}(L + U)X_n\,, \qquad X_0 \quad \text{given.}$$

Prior to implementing this algorithm, the equations are often re-ordered so as to maximise the product of the elements of D. In this way, $||\, D^{-1}(L + U)\,||$ is minimised in a straight forward way.

Gauss-Seidel algorithm The Gauss-Seidel algorithm corresponds to the choice $P = D + L$. With this choice of P, it follow that $H = (I - P^{-1}A) = -(D + L)^{-1}U$ and the iterative scheme is

$$X^{(n+1)} = (D + L)^{-1}B - (D + L)^{-1}UX^{(n)}\,, \qquad X^{(0)} \quad \text{given}\,.$$

This scheme will converge provided $(D + L)^{-1}U$ is a contraction mapping.

Example Use the Gauss-Jacobi and Gauss-Seidel algorithms to solve the system of equations

$$\begin{aligned} 8x &+ y - z &&= 8 \\ 2x &+ y + 9z &&= 12 \\ x &- 7y + 2z &&= -4\,. \end{aligned}$$

Solution The original equations are re-order to give

$$\left.\begin{aligned} 8x &+ y - z &&= 8 \\ x &- 7y + 2z &&= -4 \\ 2x &+ y + 9z &&= 12 \end{aligned}\right] \qquad \rightarrow \qquad \begin{aligned} x &= 1 - y/8 + z/8 \\ y &= 4/7 + x/7 + 2z/7 \\ z &= 4/3 - 2x/9 - y/9 \end{aligned}$$

The re-ordered equations form the basis of the Gauss-Jacobi and Gauss-Seidel iterative scheme, which in this instance are respectively

Gauss-Jacobi	**Gauss-Seidel**
$x_1^{(n+1)} = 1 - \frac{x_2^{(n)}}{8} + \frac{x_3^{(n)}}{8}$	$x_1^{(n+1)} = 1 - \frac{x_2^{(n)}}{8} + \frac{x_3^{(n)}}{8}$
$x_2^{(n+1)} = \frac{4}{7} + \frac{1}{7}x_1^{(n)} + \frac{2}{7}x_3^{(n)}$	$x_2^{(n+1)} = \frac{4}{7} + \frac{1}{7}x_1^{(n+1)} + \frac{2}{7}x_3^{(n)}$
$x_3^{(n+1)} = \frac{4}{3} - \frac{2}{9}x_1^{(n)} - \frac{x_2^{(n)}}{9}$	$x_3^{(n+1)} = \frac{4}{3} - \frac{2}{9}x_1^{(n+1)} - \frac{1}{9}x_2^{(n+1)}$

.

Both schemes take the starting values $x_1^{(0)} = x_2^{(0)} = x_3^{(0)} = 0$. Convergence is rapid with the solution

$x_1 = x_2 = x_3 = 1$ being obtained approximately after 3 iterations.

Iteration	x_1	x_2	x_3	x_1	x_2	x_3
0	0	0	0	0	0	0
1	1	$\frac{4}{7}$	$\frac{4}{3}$	1	$\frac{5}{7}$	$\frac{65}{63}$
2	$\frac{53}{56}$	$\frac{23}{21}$	$\frac{26}{27}$	$\frac{253}{252}$	$\frac{1781}{1764}$	~ 1

3.2.3 Comparison of Gauss-Jacobi and Gauss-Seidel

Although no general comparisons between the Gauss-Jacobi and the Gauss-Seidel algorithms are known, comparisons exist for some particular classes of matrices which occur often in practice. Perhaps the best known of these results is the Stein-Rosenberg theorem. The theorem applies to matrices A with the property that either each diagonal entry of A is positive and all off-diagonal entries of A are non-positive, or alternatively, each diagonal entry of A is negative and all off-diagonal entries of A are non-negative. For this class of matrices, the Gauss-Jacobi and Gauss-Seidel algorithms either both converge or both diverge. If both converge then Gauss-Seidel will converge faster than Gauss-Jacobi. Alternatively, if both diverge then Gauss-Seidel will diverge faster than Gauss-Jacobi. One point to note, however, is that Gauss-Seidel is intrinsically a serial algorithm while Gauss-Jacobi is intrinsically a parallel algorithm. The optimal choice of algorithm may therefore depend on the properties of the computing machine.

Example Determine whether or not the Gauss-Jacobi algorithm can be made to converge for the matrix

$$A = \begin{bmatrix} 1 & 3 & 0 \\ 2 & 2 & 1 \\ 0 & 1 & 2 \end{bmatrix}.$$

Solution It is necessary to compute the eigenvalues of $-D^{-1}(L+U)$, that is, solve the equation $\det(D^{-1}(L+U)+\lambda I) = 0$. Since D is non-singular then λ satisfies $\det(L+U+\lambda D) = 0$. The rows of A are re-ordered by interchanging rows 1 and 2 to maximise the likelihood that $-D^{-1}(L+U)$ is a contraction matrix. The eigenvalues λ satisfy

$$\begin{vmatrix} 2\lambda & 2 & 1 \\ 1 & 3\lambda & 0 \\ 0 & 1 & 2\lambda \end{vmatrix} = 12\lambda^3 - 4\lambda + 1 = 0\,.$$

The stationary values of this cubic function satisfy $36\lambda^2 - 4 = 0$ from which it follows that the cubic has a minimum turning point at $(1/3, 1/9)$ and a maximum turning point at $(-1/3, 17/9)$. Therefore the

cubic has exactly one real solution and it lies in the interval $(-1, -1/3)$. The remaining solutions are complex conjugate pairs. The product of the three solutions is $-1/12$ and so the complex conjugate pair solutions must lie within a circle of radius $1/2$ about the origin. Therefore all eigenvalues lie within the unit circle and so the Gauss-Jacobi algorithm based on the re-arranged matrix

$$\begin{bmatrix} 2 & 2 & 1 \\ 1 & 3 & 0 \\ 0 & 1 & 2 \end{bmatrix}$$

will converge. You should show that the Gauss-Jacobi algorithm based on the original matrix diverges.

3.2.4 Diagonal dominance

An $n \times n$ matrix A is said to be **diagonally dominated** if the sum (in modulus) of all the off-diagonal elements in any row is less than the modulus of the diagonal element in that row. Let V be a vector of dimension n, then the **supremum norm** of V is defined by

$$|| \, V \, || = \max_{1 \le j \le n} | \, V_j \, |$$

In effect, this norm measures the distance between two vectors as the modulus of the largest difference between their paired elements. If a matrix is diagonally dominated, it can be shown that the Gauss-Jacobi and Gauss-Seidel schemes are guaranteed to converge in the supremum norm. To prove these results, it is convenient to define $E = -D^{-1}L$ and $F = -D^{-1}U$ then E is a strictly lower triangular matrix and F is a strictly upper triangular matrix.

Gauss-Jacobi

Suppose A is diagonally dominated. Since $-D^{-1}(L+U) = E+F$, then it is necessary to demonstrated that $|| \, D^{-1}(L+U)V \, || = || \, (E+F)V \, || < || \, V \, ||$ for any vector V. The definition of D, L and U from A ensures that $D^{-1}A = I + D^{-1}L + D^{-1}U = I - (E+F)$. Therefore $E + F = I - D^{-1}A$ so that

$$[(E+F)V]_i = \sum_{q=1}^{n} [(E+F)]_{iq} V_q = - \sum_{q=1, q \ne i}^{n} \frac{a_{iq} \, V_q}{a_{ii}}$$

for all vectors V. Thus

$$\left| \, [(E+F)V]_i \, \right| \le \left| \sum_{q=1, q \ne i}^{n} \frac{a_{iq} V_q}{a_{ii}} \right| \le \sum_{q=1, q \ne i}^{n} \left| \frac{a_{iq}}{a_{ii}} \right| | \, V_q \, | < \max_{1 \le q \le n, q \ne i} | \, V_q \, |$$

The supremum norm is calculated by considering all values of i. Since $|| \, (E+F)V \, || < || \, V \, ||$, then it follows immediately that $(E+F)$ is a contraction mapping if A is a diagonally dominated matrix.

If diagonal dominance is weakened by allowing the sum of the absolute values of the off-diagonal elements in any row to equal the modulus of the diagonal element in that row, then it can no longer

be asserted that $D^{-1}(L+U)$ is a contraction mapping, and the eigenvalues of $D^{-1}(L+U)$ must now be calculated.

Gauss-Seidel

Suppose A is diagonally dominated. Observe first that

$$(D+L)^{-1}U = [D(I+D^{-1}L)]^{-1}U = (I+D^{-1}L)^{-1}D^{-1}U = -(I-E)^{-1}F\,.$$

It must be shown that $||\,(D+L)^{-1}UV\,|| = ||\,(I-E)^{-1}FV\,|| < ||\,V\,||$ for any vector V. Since E is a strictly lower triangular $n\times n$ matrix then $E^n = 0$ and it now follows that

$$(I-E)^{-1} = I + E + E^2 - \cdots + E^{n-1}\,.$$

The individual elements of $(I-E)^{-1}$ may be compared to get

$$\begin{aligned} |\,(I-E)^{-1}\,| &= |\,I+E+E^2+\cdots+E^{n-1}\,| \\ &< 1+|\,E\,|+|\,E\,|^2+\cdots+|\,E\,|^{n-1} \\ &= (I-|\,E\,|)^{-1}\,. \end{aligned}$$

Recall from the analysis of the Gauss-Jacobi algorithm that $(|\,E\,|+|\,F\,|)|\,V\,| \le |\,V\,|$. This inequality may be re-expressed in the form $|\,F\,||\,V\,| \le (I-|\,E\,|\,)\,|\,V\,|$ which in turn gives $(I-|\,E\,|\,)^{-1}|\,F\,||\,V\,| \le |\,V\,|$. Therefore, given any vector V, it follows that

$$||\,(I-E)^{-1}FV\,|| < ||\,V\,||\,.$$

and so $(D+L)^{-1}U$ is a contraction mapping if A is a diagonally dominated matrix.

Example Investigate the convergence of the Gauss-Seidel algorithm for the coefficient matrices

$$(i)\quad \begin{bmatrix} 3 & 1 & 1 \\ -1 & 2 & 0 \\ 1 & 2 & -4 \end{bmatrix} \qquad (ii)\quad \begin{bmatrix} 3 & 2 & 1 \\ -1 & 2 & 0 \\ 1 & 2 & -4 \end{bmatrix}.$$

Solution The first matrix exhibits strict diagonal dominance and so the Gauss-Seidel algorithm is guaranteed to converge. On the other hand, diagonal dominance is not strict for the second matrix and so it is necessary to compute the eigenvalues of $(D+L)^{-1}U$. Suppose $(D+L)^{-1}UV = \lambda V$ then $[\lambda(D+L)+U]V = 0$ and so the eigenvalues are solutions of

$$\det[\lambda(D+L)+U] = \begin{vmatrix} 3\lambda & 2 & 1 \\ -\lambda & 2\lambda & 0 \\ \lambda & 2\lambda & -4\lambda \end{vmatrix} = -12\lambda^2(2\lambda+1) = 0\,.$$

The eigenvalues are $\lambda = 0$ (twice) and $\lambda = -1/2$. Since all eigenvalues lie within the unit circle then the Gauss-Seidel algorithm converges on this occasion.

Example Construct the finite difference scheme for the solution of the partial differential equation

$$u_{xx} + u_{yy} + a(x,y)u_x + b(x,y)u_y + c(x,y)u = f$$

with Dirichlet boundary conditions. Investigate conditions under which the resulting system of linear equations will be diagonally dominated.

Solution The central difference representation of the differential equation is

$$\frac{u_{i-1,j} + u_{i,j-1} - 4u_{i,j} + u_{i,j+1} + u_{i+1,j}}{h^2} + a_{i,j}\,\frac{u_{i+1,j} - u_{i-1,j}}{2h} + b_{i,j}\,\frac{u_{i,j+1} - u_{i,j-1}}{2h} + c_{i,j}u_{i,j} = f_{i,j}\,.$$

This equation is multiplied by $2h^2$ and like terms collected together to obtain

$$(2 - ha_{i,j})u_{i-1,j} + (2 - hb_{i,j})u_{i,j-1} - (8 - 2h^2c_{i,j})u_{i,j} + (2 + hb_{i,j})u_{i,j+1} + (2 + ha_{i,j})u_{i+1,j} = 2h^2 f_{i,j}\,.$$

Strict diagonal dominance requires that

$$|\,8 - 2h^2c_{i,j}\,| > |\,2 - ha_{i,j}\,| + |\,2 - hb_{i,j}\,| + |\,2 + hb_{i,j}\,| + |\,2 + ha_{i,j}\,|\,.$$

For suitably small h, this inequality becomes

$$8 - 2h^2c_{i,j} > 2 - ha_{i,j} + 2 - hb_{i,j} + 2 + hb_{i,j} + 2 + ha_{i,j} \quad \rightarrow \quad c_{i,j} < 0\,.$$

Thus the finite difference representation of the partial differential equation will be strictly diagonally dominated provided $c(x,y) < 0$ in the region in which the solution is sought.

Example Construct the finite difference scheme for the solution of the partial differential equation

$$u_{xx} + u_{yy} = -2\,, \qquad (x,y) \in \mathcal{D}\,, \qquad u = 0 \text{ on } \partial\mathcal{D}\,,$$

where $\mathcal{D} = (-1,1) \times (-1,1)$. Iterate the finite difference scheme for chosen h, and use the exact solution

$$u(x,y) = 1 - y^2 - \frac{32}{\pi^3}\sum_{k=0}^{\infty} \frac{(-1)^k \cosh\left(\frac{(2k+1)\pi x}{2}\right)\cos\left(\frac{(2k+1)\pi y}{2}\right)}{(2k+1)^3 \cosh\left(\frac{(2k+1)\pi}{2}\right)}$$

for the boundary value problem to investigate the convergence of the finite difference scheme.

Solution The finite difference scheme to solve $u_{xx} + u_{yy} = -2$ is

$$u_{i-1,j} + u_{i,j-1} - 4u_{i,j} + u_{i,j+1} + u_{i+1,j} = -2h^2\,,$$

which may be re-expressed in the form

$$u_{i,j} = \frac{1}{4}\Big[u_{i-1,j} + u_{i,j-1} + u_{i,j+1} + u_{i+1,j}\Big] + \frac{h^2}{2} \tag{3.2}$$

with $h = 2/n$, $x_i = -1 + ih$ and $y_j = -1 + jh$. Equation (3.2) may be used as the basis of the iterative algorithm

$$u_{i,j}^{(k+1)} = \frac{1}{4}\left[u_{i-1,j}^{(k)} + u_{i,j-1}^{(k)} + u_{i,j+1}^{(k)} + u_{i+1,j}^{(k)}\right] + \frac{h^2}{2}, \qquad u_{i,j}^{(0)} = 0 \tag{3.3}$$

where the initial solution is taken to be $u = 0$, which satisfies the boundary condition. For example, when $n = 2$ then $u_{1,1}^{(1)} = u_{1,1}^{(2)} = \cdots = h^2/2$ and the iterative algorithm clearly terminates after two iterations. To control the number of Gauss-Seidel iteration, the root mean square error norm

$$E_n = \left[\frac{1}{(n+1)^2}\sum_{i,j=0}^{n}\left|u_{i,j}^{\text{new}} - u_{i,j}^{\text{old}}\right|^2\right]^{1/2} = \frac{1}{n+1}\left[\sum_{i,j=0}^{n}\left|u_{i,j}^{\text{new}} - u_{i,j}^{\text{old}}\right|^2\right]^{1/2}$$

is used to terminate Gauss-Seidel iteration whenever $E_n < 10^{-6}$. The result of this procedure, now measuring the numerical error against the true solution, is given in Table 3.1.

n	ITER	E_n	E_{n-1}/E_n
2	2	0.0893708	–
4	19	0.0207940	4.298
8	70	0.0048225	4.312
16	245	0.0011738	4.109
32	837	0.0003828	3.067
64	2770	0.0004880	0.784
128	8779	0.0016882	0.289

Table 3.1:
Comparison of L^2 errors in the solution of $u_{xx} + u_{yy} = -2$ when using Gauss-Seidel iteration and the termination condition $E_n < 10^{-6}$.

Clearly the Gauss-Seidel algorithm is using many more iterations to produce an inferior result as the mesh is refined by increasing n. This is a counter-intuitive result. The difficulty lies in the stopping criterion. The use of an absolute termination condition can be problematic when an iterative procedure converges very slowly as happens here. The usual way to ameliorate this hazard is to use a termination condition which becomes increasingly demanding as the number of iterations increases. One popular choice is to make the termination condition inversely proportional to the number of iterations performed. The same calculation repeated with the termination condition $E_n < 10^{-5}/ITER$, where $ITER$ counts the number of iterations of the Gauss-Seidel algorithm, gives

n	ITER	E_n	E_{n-1}/E_n	$ITER/n^2$
2	2	0.0893708	–	0.500
4	20	0.0207934	4.298	1.250
8	83	0.0048172	4.316	1.297
16	336	0.0011487	4.194	1.313
32	1344	0.0002802	4.100	1.313
64	5379	0.0000696	4.024	1.313
128	21516	0.0000178	3.901	1.313

Table 3.2:
Comparison of L^2 errors in the solution of $u_{xx} + u_{yy} = -2$ when using Gauss-Seidel iteration and the termination condition $E_n < 10^{-5}/ITER$.

The fact that $ITER/n^2$ is effectively constant is simply another manifestation of the fact that the numerical accuracy is proportional to h^2, that is $1/n^2$.

Multi-grid procedure

The single grid procedure can be improved by using a multi-grid approach in which the mesh is progressively refined by halving h, or equivalently, doubling n. The solution for n can be used as the starting configuration for an iteration based on $2n$. Values of the solution at the intermediate points introduced by the multi-grid approach can be obtained by linear interpolation of the existing grid points. Specifically

$$u_{2i+1,j} = \frac{u_{2i,j} + u_{2i+2,j}}{2}\,, \qquad u_{i,2j+1} = \frac{u_{i,2j} + u_{i,2j+2}}{2}\,.$$

Table 3.3 illustrates the use of this procedure for the problem under discussion.

n	ITER	E_n	E_{n-1}/E_n	$ITER/n^2$
2	2	0.0893708	–	0.500
4	20	0.0207933	4.298	1.250
8	80	0.0048171	4.317	1.250
16	316	0.0011487	4.193	1.234
32	1252	0.0002802	4.100	1.223
64	4989	0.0000697	4.022	1.218
128	19923	0.0000179	3.892	1.216

Table 3.3:
Comparison of L^2 errors in the solution of $u_{xx} + u_{yy} = -2$ when using Gauss-Seidel iteration, a multi-grid procedure and the termination condition $|\text{Error}| < 10^{-5}/ITER$.

3.3 Non-rectangular domains

The finite difference procedure can be used directly to solve the Dirichlet boundary value problem on irregular domains with boundary curves formed by connecting line segments oriented at right angles to each other, for example, an "I" beam. Other non-rectilinear regions can also be treated easily by the finite difference procedure if their boundary curves are coordinate lines. For example, equations expressed in polar coordinates over the interior of a circle. Laplace's equation in polar coordinates is

$$\frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r} + \frac{1}{r^2}\frac{\partial^2 u}{\partial \theta^2} = f(r,\theta)\,. \tag{3.4}$$

This equation may be solved using the finite difference procedure by taking $r_i = ih$ and $\theta_j = jk$ in which $h = R/n$ and $k = 2\pi/m$. Each derivative appearing in equation (3.4) may be replaced by its finite difference representation to get

$$\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} + \frac{1}{r_i}\frac{u_{i+1,j} - u_{i-1,j}}{2h} + \frac{1}{r_i^2}\frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{k^2} = f_{i,j}$$

except at the origin. The finite difference equation contributed by the origin may be constructed by returning to the original cartesian version of the Laplace molecule. This molecule can be used provided nodes of θ fall on the coordinate axes, that is, when m is a multiple of 4.

Finally, domains that exhibit little or no regularity are most effectively treated using finite element methods.

Example (May 2001) Consider the boundary value problem

$$u_{xx} + u_{yy} = -2\,, \qquad (x,y) \in \mathcal{D}$$

where $\mathcal{D}$ is the interior of the rectangle with vertices at $(0,0)$, $(0,1)$, $(1/2.0)$ and $(1/2,1)$, and the solution is required to satisfy the boundary conditions

$$u(x,0) = 1\,, \qquad u(x,1) = 0\,, \qquad u(0,y) = 1-y\,, \qquad u_x(1/2.y) = u^2(1/2,y)\,.$$

(i) Construct a finite difference representation of this problem for general h using the fictitious nodes procedure to describe the gradient boundary condition.

(ii) Write out the specific equations to be solved in the case in which $h = 1/4$.

(iii) Restructure these equations in a form suitable for iteration.

Solution

(i) Take $x_i = ih$ and $y_j = jh$ where $h = 1/2n$. In this case $i = 0, 1, \cdots, n$ and $j = 0, 1\cdots, 2n$. The Laplace molecule gives

$$u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1} - 4u_{i,j} = -2h^2$$

where $i = 1, \cdots, (n-1)$ and $j = 1, \cdots, (2n-1)$. The boundary conditions yield

$$\begin{aligned} u_{0,j} &= 1 - jh & j &= 0, \cdots, 2n \\ u_{i,0} &= 1 & i &= 0, \cdots, n \\ u_{i,2N} &= 0 & i &= 0, \cdots, n \end{aligned}$$

The boundary condition on $x = 1/2$ requires more work. The fictitious nodes procedure gives

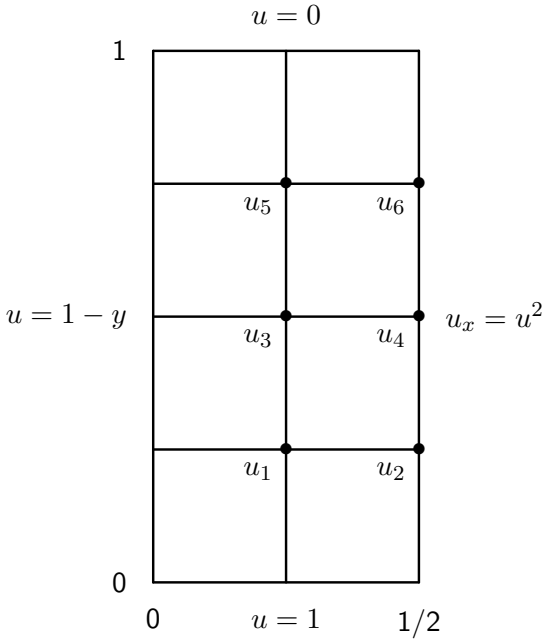
$$\left.\begin{aligned} u_{n+1,j} - u_{n-1,j} &= 2hu_{n,j}^2 \\ u_{n-1,j} + u_{n+1,j} + u_{n,j-1} + u_{n,j+1} - 4u_{n,j} &= -2h^2 \end{aligned}\right] \quad j = 1, \cdots, 2n-1\,.$$

The fictitious value $u_{n+1,j}$ is now eliminated between these equations to obtain

$$2u_{n-1,j} + 2hu_{n,j}^2 + u_{n,j-1} + u_{n,j+1} - 4u_{n,j} = -2h^2 \quad j = 1, \cdots, 2n-1\,.$$

This is the final form for the boundary condition on $x = 1/2$.

(ii) Now take $h = 1/4$ or $n = 2$. Let $u_1, \cdots, u_6$ be the unknowns defined in the diagram



The finite difference for $h = 1/4$ are therefore

$$\left.\begin{array}{rcl} u_2 + \frac{3}{4} + 1 + u_3 - 4u_1 &=& -\frac{1}{8} \\ 2u_1 + 2hu_{2,j}^2 + 1 + u_4 - 4u_2 &=& -\frac{1}{8} \\ u_4 + \frac{1}{2} + u_1 + u_5 - 4u_3 &=& -\frac{1}{8} \\ 2u_3 + 2hu_4^2 + u_2 + u_6 - 4u_4 &=& -\frac{1}{8} \\ u_6 + \frac{1}{4} + u_3 + 0 - 4u_5 &=& -\frac{1}{8} \\ 2u_5 + 2hu_6^2 + u_4 - 4u_6 &=& -\frac{1}{8} \end{array}\right] \quad \rightarrow \quad \begin{array}{rcl} u_2 + u_3 - 4u_1 &=& -\frac{15}{8} \\ 4u_1 + u_2^2 + 2u_4 - 8u_2 &=& -\frac{9}{4} \\ u_4 + u_1 + u_5 - 4u_3 &=& -\frac{5}{8} \\ 4u_3 + u_4^2 + 2u_2 + 2u_6 - 8u_4 &=& -\frac{1}{4} \\ u_6 + u_3 - 4u_5 &=& -\frac{3}{8} \\ 4u_5 + u_6^2 + 2u_4 - 8u_6 &=& -\frac{1}{4} \end{array}$$

(iii) The j^{th} equation is solved for u_j to get the basic formulation of the numerical problem, which is subsequently rewritten in iterative forma and leads to the Gauss-Seidel algorithm

$$\left.\begin{array}{rcl} u_1 &=& \frac{1}{4}\left(u_2 + u_3 + \frac{15}{8}\right) \\ u_2 &=& \frac{2}{8-u_2}\left(2u_1 + u_4 + \frac{9}{8}\right) \\ u_3 &=& \frac{1}{4}\left(u_1 + u_4 + u_5 + \frac{5}{8}\right) \\ u_4 &=& \frac{2}{8-u_4}\left(u_2 + 2u_3 + u_6 + \frac{1}{8}\right) \\ u_5 &=& \frac{1}{4}\left(u_3 + u_6 + \frac{3}{8}\right) \\ u_6 &=& \frac{2}{8-u_6}\left(u_4 + 2u_5 + \frac{1}{8}\right) \end{array}\right] \quad \rightarrow \quad \begin{array}{rcl} u_1^{(k+1)} &=& \frac{1}{4}\left(u_2^{(k)} + u_3^{(k)} + \frac{15}{8}\right) \\ u_2^{(k+1)} &=& \frac{2}{8-u_2^{(k)}}\left(2u_1^{(k+1)} + u_4^{(k)} + \frac{9}{8}\right) \\ u_3^{(k+1)} &=& \frac{1}{4}\left(u_1^{(k+1)} + u_4^{(k)} + u_5^{(k)} + \frac{5}{8}\right) \\ u_4^{(k+1)} &=& \frac{2}{8-u_4^{(k)}}\left(u_2^{(k+1)} + 2u_3^{(k+1)} + u_6^{(k)} + \frac{1}{8}\right) \\ u_5^{(k+1)} &=& \frac{1}{4}\left(u_3^{(k+1)} + u_6^{(k)} + \frac{3}{8}\right) \\ u_6^{(k+1)} &=& \frac{2}{8-u_6^{(k)}}\left(u_4^{(k+1)} + 2u_5^{(k+1)} + \frac{1}{8}\right) \end{array}$$

Exercises

Question 19. Two less common molecules for the computation of $u_{xx} + u_{yy}$ are

$$\text{(a)} \quad \frac{1}{2h^2}\begin{bmatrix} 1 & 0 & 1 \\ 0 & -4 & 0 \\ 1 & 0 & 1 \end{bmatrix} \qquad \text{(b)} \quad \frac{1}{6h^2}\begin{bmatrix} 1 & 4 & 1 \\ 4 & -20 & 4 \\ 1 & 4 & 1 \end{bmatrix}.$$

Determine the order of the error term associated with each of these molecules. Does this change if the scheme is used to solve Laplace's equation and not a more general partial differential equation of which the Laplacian is a component part.

Question 20. Prove that Jacobi iteration is convergent but Gauss-Seidel iteration is divergent for the equations

$$\begin{bmatrix} 1 & 2 & 4 \\ 1/8 & 1 & 1 \\ -1 & 4 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \\ 7 \end{bmatrix}.$$

Question 21. The system of equations

$$\begin{bmatrix} 3 & 2 & 1 \\ 2 & 3 & 2 \\ 1 & 2 & 3 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 6 \\ 7 \\ 6 \end{bmatrix},$$

is not diagonally dominant. Calculate the spectral radius of the iteration matrix for both the Gauss-Jacobi and the Gauss-Seidel algorithms. What can you say about the convergence of these procedures?

Question 22. The coefficient matrix of a system of linear equations $AX = B$ is defined by

$$A = \begin{bmatrix} 1 & c & 1 \\ c & 1 & c \\ -c^2 & c & 1 \end{bmatrix},$$

where c is real and non-zero. Find the range of values for which Gauss-Seidel iteration converges. Will the Jacobi iteration converge when $c = 2$?

Question 23. Prove that pivoting is unnecessary in the LU factorisation of A if A^{T} is diagonally dominated. To establish this result you may find it useful to establish the block matrix identity

$$\begin{bmatrix} \alpha & W^{\mathrm{T}} \\ V & C \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \dfrac{V}{\alpha} & I \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 0 & C - \dfrac{VW^{\mathrm{T}}}{\alpha} \end{bmatrix}\begin{bmatrix} \alpha & W^{\mathrm{T}} \\ 0 & I \end{bmatrix}$$

Question 24. Let A be a symmetric and positive definite $n \times n$ matrix.

(a) Show that each diagonal entry of A is positive.

(b) Show that $H = (D + L)^{-1}U$ satisfies the property

$$D^{1/2} H D^{-1/2} = (I + L_1)^{-1} L_1^{\mathrm{T}}$$

where L_1 is a strictly lower triangular matrix to be identified and D and L are defined from A in the usual way.

(c) Show further that

$$D^{-1/2} A D^{-1/2} = I + L_1 + L_1^{\mathrm{T}} .$$

(d) By considering the eigenvalues of H_1, show that the Gauss-Seidel algorithm converges for positive definite matrices.

Question 25. The torsion of an elastic beam of square cross-section requires the solution of the boundary value problem

$$\Phi_{xx} + \Phi_{yy} + 2 = 0, \quad (x, y) \in (-1, 1) \times (-1, 1),$$

with $\Phi = 0$ on the boundary of the square. First, write out the discretised scheme using a step length $h = 1/2$. Now use the symmetry of the problem to reduce the number of unknowns to three. Finally, solve the equations to show that $\Phi(0, 0) \approx 0.562$ [exact solution is 0.589].

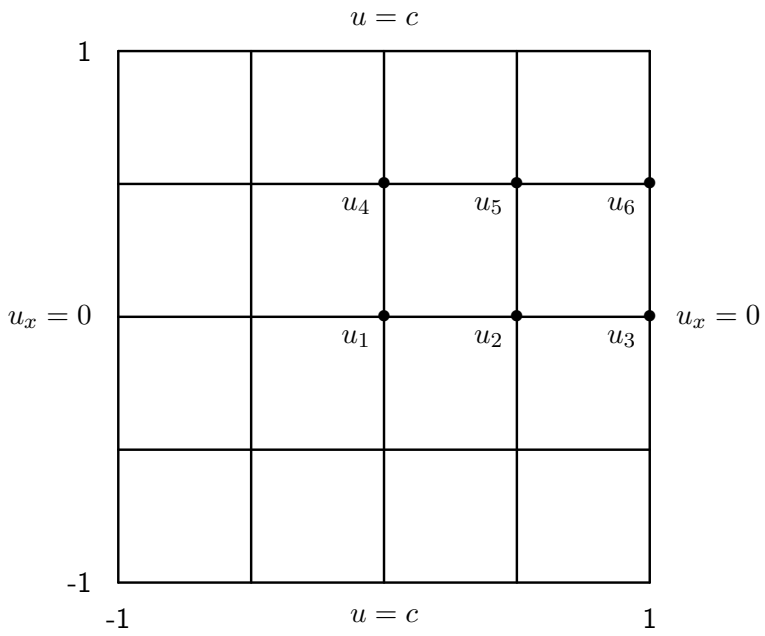
Question 26. (May 1999) Consider the equation

$$u_{xx} + u_{yy} = u^3 , \qquad (x, y) \in \mathcal{D}$$

where $\mathcal{D}$ is the interior of the square with vertices $(1, 1), (1, -1), (-1, 1)$ and $(-1, -1)$, and u is required to satisfy the boundary conditions

$$u_x = 0 \quad x = \pm 1 , \qquad u = c \quad y = \pm 1$$

where c is a constant. You may take advantage of symmetry and assume that the solution is an even function in x and y. Use this information to discretise the partial differential equations and obtain equations for $u_1, \cdots, u_6$ as illustrated in the diagram.



You may again implement the gradient boundary conditions by either a backward difference approximation of appropriate order, or by introducing fictitious nodes beyond $x = 1$

Having obtained the (nonlinear) equations for the unknowns $u_1, \cdots, u_6$, devise an iterative scheme to solve them. Ensure that your scheme is as robust as possible, that is, try to ensure that it will converge for any initial guess and for any choice of c.

Chapter 4

Parabolic equations

4.1 Introduction

The canonical parabolic equation in one dimension is the diffusion equation

$$u_t = u_{xx}\,, \qquad (x,t) \in (0,1) \times (0,\infty)$$

with initial condition $u(x,0) = f(x)$ and boundary conditions

$$u(0,t) = g_0(t)\,, \qquad u(1,t) = g_1(t)\,.$$

By contrast with elliptic equations where the same step length could often be used in each direction, this is impossible in a parabolic equation since time and space are different physical variables, and therefore cannot be equated. Let $x_i = ih$ be a dissection of $[0,1]$ with $h = 1/n$, then the finite difference representation of the diffusion equation is

$$\frac{du_i}{dt} = \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2}\,, \qquad u_0 = g_0(t)\,, \quad u_n = g_1(t)\,.$$

Thus the solution of the diffusion equation is constructed with respect to time by integrating a family of $(n-1)$ ordinary differential equations in which the i^{th} unknown is the time course of $u(x_i,t)$. These equations are solved with the initial condition $u_i(0) = f(x_i)$. In order to proceed further, it is necessary to develop efficient procedures to integrate large families of ordinary differential equations.

4.1.1 Euler's method

Let $y(t)$ be the solution of the initial value problem

$$\frac{dy}{dt} = f(y,t)\,, \quad t > 0\,, \qquad y(0) = y_0\,.$$

Let $t_j = jk$ where k is the discretisation of time and let $y_j = y(t_j)$, then Euler's method asserts that

$$y_{j+1} = y_j + kf(y_j, t_j)\,.$$

The scheme is derived from the Taylor series expansion

$$y(t+k) = y(t) + k\frac{dy(t)}{dt} + \frac{k^2}{2}\frac{d^2y(t)}{dt^2} + O(k^3)$$

by replacing dy/dt with $f(y,t)$ and ignoring subsequent terms. Euler's scheme is an **explicit** algorithm because $y(t+k)$ appears only on **one** side of the scheme. Let $u_{i,j} = u(x_i, t_j)$, then Euler's scheme applied to the diffusion equation gives

$$u_{i,j+1} = u_{i,j} + r\,(u_{i+1,j} - 2u_{i,j} + u_{i-1,j}\,)$$

where $r = k/h^2$ is called the *Courant* Number. This finite difference scheme may be rewritten

$$u_{i,j+1} - r\,u_{i+1,j} - (1-2r)\,u_{i,j} - r\,u_{i-1,j} = 0$$

which in turn leads to the computational molecule

$$u_t - u_{xx} = \boxed{\begin{array}{ccc} 0 & 1 & 0 \\ -r & -(1-2r) & -r \\ 0 & 0 & 0 \end{array}}$$

Example

Solve the diffusion equation $u_t = u_{xx}$ where $x \in (0,1)$ with initial condition $u(x,0) = \sin(2\pi x)$ and boundary conditions $u(0,t) = u(1,t) = 0$. The exact solution is $u(x,t) = e^{-4\pi^2 t}\sin(2\pi x)$

Solution

		k values for $n=8$			k values for $n=16$			k values for $n=32$		
t	$u(1/4,t)$	10^{-2}	10^{-3}	10^{-4}	10^{-2}	10^{-3}	10^{-4}	10^{-2}	10^{-3}	10^{-4}
0.01	0.67	0.63	0.68	0.69	0.56	0.62	0.63	0.59	0.66	0.66
0.02	0.45	0.39	0.47	0.47	0.34	0.42	0.42	0.36	0.44	0.45
0.03	0.31	0.24	0.32	0.32	0.21	0.28	0.29	0.22	0.29	0.30
0.04	0.21	0.15	0.22	0.22	0.13	0.19	0.19	0.13	−23.6	0.20
0.05	0.14	0.10	0.15	0.15	0.08	0.13	0.13	0.08	∗∗∗	0.14
0.06	0.09	0.06	0.10	0.11	0.05	0.09	0.09	0.05	∗∗∗	0.09
0.07	0.06	0.04	0.07	0.07	0.03	0.06	0.06	0.03	∗∗∗	0.06
0.08	0.04	0.02	0.05	0.05	0.02	0.04	0.04	0.02	∗∗∗	0.04
0.09	0.03	0.01	0.03	0.03	0.01	0.03	0.03	0.01	∗∗∗	0.03
0.10	0.02	0.01	0.02	0.02	0.01	0.02	0.02	−0.03	∗∗∗	0.02
0.11	0.01	0.01	0.01	0.02	0.00	0.01	0.01	1.54	∗∗∗	0.01
0.12	0.01	0.00	0.01	0.01	0.00	0.01	0.01	−59.6	∗∗∗	0.01
0.13	0.01	0.00	0.01	0.01	0.00	0.01	0.01	∗∗∗	∗∗∗	0.01
0.14	0.00	0.00	0.00	0.01	0.00	0.00	0.00	∗∗∗	∗∗∗	0.00
0.15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	∗∗∗	∗∗∗	0.00

Note how the efficacy of the method depends on the size of r - small h entails even smaller k.

4.1.2 Richardson's method

Richardson's method replaces the time derivative by its central difference formula

$$\frac{du_i}{dt} = \frac{u_{i,j+1} - u_{i,j-1}}{2k} + O(k^2)\,.$$

The result is the Richardson finite difference scheme

$$u_{i,j+1} - u_{i,j-1} = \frac{2k}{h^2}\left(u_{i+1,j} - 2u_{i,j} + u_{i-1,j}\right) = 2r\left(u_{i+1,j} - 2u_{i,j} + u_{i-1,j}\right)$$

with computational molecule

$$u_t - u_{xx} = \boxed{\begin{array}{ccc} 0 & 1 & 0 \\ -2r & 4r & -2r \\ 0 & -1 & 0 \end{array}}$$

One difficulty with the Richardson scheme arises at $j = 0$ since the backward solution $u_{i,-1}$ does not exist. One way around this difficulty is to use the Euler scheme for the first step of the Richardson scheme. To investigate the effectiveness of the Richardson algorithm, the scheme is started with $j = 1$. The initial condition is used to obtain values of $u_{i,0}$ and the exact solution is used to get values for $u_{i,1}$. Values for $u_{i,j}$ with $j > 1$ are obtained by iterating the scheme. This arrangement overstates the effectiveness of the Richardson scheme since the solutions at $j = 0$ and $j = 1$ are exact.

		k values for $n = 8$			k values for $n = 16$			k values for $n = 32$		
t	$u(1/4, t)$	10^{-2}	10^{-3}	10^{-4}	10^{-2}	10^{-3}	10^{-4}	10^{-2}	10^{-3}	10^{-4}
0.01	0.67	0.49	0.66	0.68	0.44	0.60	0.62	0.46	0.64	$***$
0.02	0.45	0.30	0.45	0.47	0.28	0.41	0.42	0.30	−23.4	$***$
0.03	0.31	0.27	0.31	0.32	0.22	0.28	0.29	0.23	$***$	$***$
0.04	0.21	0.10	0.21	0.22	0.11	0.16	2.08	0.12	$***$	$***$
0.05	0.14	0.19	0.14	0.15	0.13	$***$	$***$	0.13	$***$	$***$
0.06	0.09	−0.04	0.09	0.10	0.00	$***$	$***$	0.02	$***$	$***$
0.07	0.06	0.22	0.06	0.07	0.13	$***$	$***$	0.12	$***$	$***$
0.08	0.04	−0.21	0.03	0.05	−0.10	$***$	$***$	−0.09	$***$	$***$
0.09	0.03	0.38	0.00	0.03	0.21	$***$	$***$	1.14	$***$	$***$
0.10	0.02	−0.49	−0.02	0.02	−0.26	$***$	$***$	−73.7	$***$	$***$
0.11	0.01	0.75	−0.05	0.01	0.40	$***$	$***$	$***$	$***$	$***$
0.12	0.01	−1.05	−0.08	0.00	−0.46	$***$	$***$	$***$	$***$	$***$
0.13	0.01	1.53	−0.12	−0.01	−1.62	$***$	$***$	$***$	$***$	$***$
0.14	0.00	−2.20	−0.19	−0.01	48.1	$***$	$***$	$***$	$***$	$***$
0.15	0.00	3.18	−0.34	−0.02	$***$	$***$	$***$	$***$	$***$	$***$

Table 4.1: Calculations illustrating the use of Richardson's algorithm to solve $u_t = u_{xx}$ with initial condition $u(x, 0) = \sin(2\pi x)$ and boundary conditions $u(0, t) = u(1, t) = 0$.

Table 4.1 illustrates the use of Richardson's algorithm to solve the initial boundary problem discussed previously. Richardson's algorithm is unstable for the choices of h and k in this table. In fact, it will be shown later that the algorithm is unstable for all choices of h and k. This result, however, is not obvious *a priori* and is counter-intuitive in the respect that the treatment of the time derivative is more accurate than the Euler scheme and so one would expect the Richardson algorithm to be superior to the Euler algorithm.

4.1.3 Dufort-Frankel method

The Dufort-Frankel method is a modification of the Richardson method in which $u_{i,j}$ is replaced by its time averaged value, that is,

$$u_{i,j} = \frac{u_{i,j+1} + u_{i,j-1}}{2} + O(k^2)$$

With this modification of the Richardson algorithm, we get the Dufort-Frankel algorithm

$$\frac{u_{i,j+1} - u_{i,j-1}}{2k} = \frac{1}{h^2}\left[u_{i+1,j} - 2 \times \frac{1}{2}\left(u_{i,j+1} + u_{i,j-1}\right) + u_{i-1,j} \right].$$

Taking $r = k/h^2$, the Dufort-Frankel algorithm becomes

$$(1+2r)u_{i,j+1} - (1-2r)u_{i,j-1} - 2r\, u_{i+1,j} - 2r u_{i-1,j} = 0$$

with computational molecule

$$u_t - u_{xx} = \boxed{\begin{array}{ccc} 0 & 1+2r & 0 \\ -2r & 0 & -2r \\ 0 & -1+2r & 0 \end{array}}$$

Table 4.2 illustrates the result of using the Dufort-Frankel algorithm to solve the original partial differential equation by iterating

$$u_{i,j+1} = \frac{1-2r}{1+2r}\, u_{i,j-1} + \frac{2r}{1+2r}\, u_{i+1,j} + \frac{2r}{1+2r}\, u_{i-1,j}$$

taking $u_{i,0}$ from the boundary condition and $u_{i,1}$ from the exact solution. The numerical scheme is clearly more stable with no evidence of numerical blow-up.

		k values for $n = 8$			k values for $n = 16$			k values for $n = 32$		
t	$u(1/4, t)$	10^{-2}	10^{-3}	10^{-4}	10^{-2}	10^{-3}	10^{-4}	10^{-2}	10^{-3}	10^{-4}
0.01	0.67	0.41	0.66	0.68	0.34	0.60	0.62	0.35	0.63	0.66
0.02	0.45	0.24	0.45	0.47	0.11	0.40	0.42	0.05	0.42	0.44
0.03	0.31	0.14	0.31	0.32	−0.06	0.27	0.29	−0.22	0.28	0.30
0.04	0.21	0.08	0.21	0.22	−0.17	0.18	0.19	−0.46	0.18	0.20
0.05	0.14	0.05	0.15	0.15	−0.22	0.12	0.13	−0.66	0.12	0.14
0.06	0.09	0.03	0.10	0.11	−0.23	0.08	0.09	−0.82	0.08	0.09
0.07	0.06	0.02	0.07	0.07	−0.20	0.06	0.06	−0.93	0.05	0.06
0.08	0.04	0.01	0.05	0.05	−0.16	0.04	0.04	−1.00	0.04	0.04
0.09	0.03	0.01	0.03	0.03	−0.11	0.03	0.03	−1.02	0.02	0.03
0.10	0.02	0.00	0.02	0.02	−0.06	0.02	0.02	−1.01	0.02	0.02
0.11	0.01	0.00	0.02	0.02	−0.02	0.01	0.01	−0.96	0.01	0.01
0.12	0.01	0.00	0.01	0.01	0.01	0.01	0.01	−0.88	0.01	0.01
0.13	0.01	0.00	0.01	0.01	0.03	0.01	0.01	−0.77	0.00	0.01
0.14	0.00	0.00	0.00	0.01	0.03	0.00	0.00	−0.65	0.00	0.00
0.15	0.00	0.00	0.00	0.00	0.04	0.00	0.00	−0.51	0.00	0.00

Table 4.2: Calculations illustrating the use of Dufort-Frankel algorithm to solve $u_t = u_{xx}$ with initial condition $u(x, 0) = \sin(2\pi x)$ and boundary conditions $u(0, t) = u(1, t) = 0$.

However, it would appear that the process of increasing n does not in itself ensure a more accurate solution. Spatial refinement cannot be achieved without an accompanying temporal refinement. Table 4.3 repeats some of these calculations with a wide range of spatial discretisations. The most important point to observe from this Table is that the scheme appears to converge as n increases, but the convergence is not to the true solution of the partial differential equation.

		$n = 32$	$n = 320$	$n = 3200$
t	$u(1/4, t)$	$k = 10^{-2}$	$k = 10^{-3}$	$k = 10^{-4}$
0.01	0.6738	0.3477	0.5742	0.6021
0.02	0.4540	0.0218	0.1873	0.2083
0.03	0.3059	−0.3038	−0.1993	−0.1853
0.04	0.2062	−0.6290	−0.5854	−0.5783
0.05	0.1389	−0.9537	−0.9710	−0.9708

Table 4.3: The exact and numerical values of $u(1/4, t)$ for the solution of the diffusion equation $u_t = u_{xx}$ with initial condition $u(x, 0) = \sin(2\pi x)$ and boundary conditions $u(0, t) = u(1, t) = 0$ are shown.

This is the worst possible scenario. Without knowing the exact solution, one might (erroneously)

take the result of this numerical calculation to be a reasonable approximation to the exact solution of the partial differential equation.

4.1.4 Crank-Nicolson method

The Crank-Nicolson algorithm averages not only $u_{i,j}$ but also $u_{i+1,j}$ and $u_{i-1,j}$. Therefore the Crank-Nicolson formulation of the diffusion equation is

$$\frac{u_{i,j+1}-u_{i,j-1}}{2k}=\frac{1}{2h^2}\left[\frac{u_{i+1,j+1}+u_{i-1,j-1}-2(u_{i,j+1}+u_{i,j-1})+u_{i-1,j+1}+u_{i-1,j-1}}{}\right].$$

This equation can be re-expressed in the form

$$\begin{aligned}\frac{u_{i,(j-1)+2}-u_{i,(j-1)}}{2k} &= \frac{1}{2h^2}\Big[u_{i+1,(j-1)+2}+u_{i-1,(j-1)}\\ &\qquad -2(u_{i,(j-1)+2}+u_{i,(j-1)})+u_{i-1,(j-1)+2}+u_{i-1,(j-1)}\Big].\end{aligned}$$

In particular, the value of the solution at time t_j nowhere enters the algorithm which in practice advances the solution from t_{j-1} to t_{j+1} through a time step of length $2k$. By reinterpreting k to be $2k$, the Crank-Nicolson algorithm becomes

$$\frac{u_{i,j+1}-u_{i,j}}{k}=\frac{1}{2h^2}\Big[u_{i+1,j+1}+u_{i+1,j}-2(u_{i,j+1}+u_{i,j})+u_{i-1,j+1}+u_{i-1,j}\Big].$$

The Courant number $r=k/h^2$ is now introduced to obtain

$$u_{i,j+1}-u_{i,j}=\frac{r}{2}\Big[u_{i+1,j+1}+u_{i+1,j}-2(u_{i,j+1}+u_{i,j})+u_{i-1,j+1}+u_{i-1,j}\Big].$$

which may be re-arranged to give

$$-\frac{r}{2}u_{i+1,j+1}+(1+r)\,u_{i,j+1}-\frac{r}{2}u_{i-1,j+1}=\frac{r}{2}u_{i+1,j}+(1-r)\,u_{i,j}+\frac{r}{2}u_{i-1,j}\,.$$

The computational module corresponding to the Crank-Nicolson algorithm is therefore

$$u_t-u_{xx}=\boxed{\begin{array}{ccc}-r/2 & 1+r & -r/2\\ -r/2 & -1+r & -r/2\\ 0 & 0 & 0\end{array}}.$$

Let $U^{(j)}=(u_{0,j},u_{1,j},\cdots,u_{n,j})$ be the vector of u values at time t_j then the Crank-Nicolson algorithm for the solution of the diffusion equation $u_t=u_{xx}$ with Dirichlet boundary conditions may be expressed in the form

$$T_{\mathrm{L}}U^{(j+1)}=T_{\mathrm{R}}U^{(j)}+F^{(j)}$$

in which T_{L} and T_{R} are tri-diagonal matrices. The first and last rows of these matrices contain the boundary conditions which may also arise in the expression for $F^{(j)}$. The initial solution is determined from the initial conditions and subsequent solutions are obtained by solving a tri-diagonal system of equations. Table 4.4 gives the solution of the original problem using the Crank-Nicolson algorithm.

		k values for $n = 8$			k values for $n = 16$			k values for $n = 32$		
t	$u(1/4, t)$	10^{-2}	10^{-3}	10^{-4}	10^{-2}	10^{-3}	10^{-4}	10^{-2}	10^{-3}	10^{-4}
0.01	0.67	0.68	0.69	0.69	0.62	0.63	0.63	0.66	0.66	0.66
0.02	0.45	0.47	0.47	0.47	0.42	0.42	0.42	0.44	0.45	0.45
0.03	0.31	0.32	0.32	0.32	0.28	0.29	0.29	0.30	0.30	0.30
0.04	0.21	0.22	0.22	0.22	0.19	0.19	0.19	0.20	0.20	0.20
0.05	0.14	0.15	0.15	0.15	0.13	0.13	0.13	0.13	0.14	0.14
0.06	0.09	0.10	0.11	0.11	0.09	0.09	0.09	0.09	0.09	0.09
0.07	0.06	0.07	0.07	0.07	0.06	0.06	0.06	0.06	0.06	0.06
0.08	0.04	0.05	0.05	0.05	0.04	0.04	0.04	0.04	0.04	0.04
0.09	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
0.10	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02
0.11	0.01	0.02	0.02	0.02	0.01	0.01	0.01	0.01	0.01	0.01
0.12	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
0.13	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
0.14	0.00	0.00	0.01	0.01	0.00	0.00	0.00	0.00	0.00	0.00
0.15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 4.4: The Crank Nicolson algorithm is used to calculate the numerical solution of the diffusion equation $u_t = u_{xx}$ with initial condition $u(x, 0) = \sin(2\pi x)$ and boundary conditions $u(0, t) = u(1, t) = 0$.

Table 4.5 illustrate the convergence of this solution as spatial resolution is refined.

		$n = 32$	$n = 320$	$n = 3200$
t	$u(1/4, t)$	$k = 10^{-2}$	$k = 10^{-3}$	$k = 10^{-4}$
0.01	0.67382545	0.67030550	0.67379098	0.67382511
0.02	0.45404074	0.44930947	0.45399428	0.45404027
0.03	0.30594421	0.30117461	0.30589725	0.30594374
0.04	0.20615299	0.20187900	0.20611081	0.20615257
0.05	0.13891113	0.13532060	0.13887560	0.13891078
0.06	0.09360186	0.09070614	0.09357313	0.09360157
0.09	0.02863695	0.02731839	0.02862376	0.02863681
0.12	0.00876131	0.00822760	0.00875593	0.00876125
0.15	0.00268047	0.00247795	0.00267842	0.00268045

Table 4.5: Calculations illustrating the convergence of the Crank Nicolson algorithm for various combinations of parameters.

4.1.5 Summary

Several numerical schemes have been examined with varying degrees of success. The performance of each scheme is now summarised.

Euler	Converges if k is sufficiently small but otherwise will blow-up.
Richardson	Seems to be unstable for all choices of the Courant number r.
Dufort-Frankel	Converges but not necessarily to the correct solution!
Crank-Nicolson	Seems to exhibit stable behaviour for all choices of the Courant number and converges to the correct solution as spatial resolution is improved.

4.2 Numerical consistency

The properties required by a numerical scheme to enable it to solve a partial differential equation can be summarised in terms of the consistency, stability and convergence of the scheme.

A scheme is said to be **consistent** if it solves the problem it purports to solve. In the context of the diffusion equation, the numerical scheme must be consistent with the partial differential equation $u_t = u_{xx}$. It would appear, for example, that the Dufort-Frankel algorithm is not consistent.

A numerical algorithm is said to be **stable** provided small errors in arithmetic remain bounded. For example, particular combinations of temporal and spatial discretisation may cause the scheme to blow up as happens with the Euler scheme. It would appear that there are no combinations of h and k for which the Richardson scheme is stable. On the other hand, calculation would suggest that the Crank-Nicolson scheme is stable for all combinations of temporal and spatial discretisation.

A scheme **converges** if it is both consistent and stable as temporal and spatial discretisation is reduced.

4.2.1 Consistency

The Dufort-Frankel and Crank-Nicolson algorithms are examined for consistency. In both cases, the analysis takes advantage of the Taylor series expansions

$$\begin{aligned}
u_{i,j+1} &= u_{i,j} + k\frac{\partial u_{i,j}}{\partial t} + \frac{k^2}{2}\frac{\partial^2 u_{i,j}}{\partial t^2} + O(k^3) \\
u_{i,j-1} &= u_{i,j} - k\frac{\partial u_{i,j}}{\partial t} + \frac{k^2}{2}\frac{\partial^2 u_{i,j}}{\partial t^2} + O(k^3) \\
u_{i+1,j} &= u_{i,j} + h\frac{\partial u_{i,j}}{\partial x} + \frac{h^2}{2}\frac{\partial^2 u_{i,j}}{\partial x^2} + \frac{h^3}{6}\frac{\partial^3 u_{i,j}}{\partial x^3} + O(h^4) \\
u_{i-1,j} &= u_{i,j} - h\frac{\partial u_{i,j}}{\partial x} + \frac{h^2}{2}\frac{\partial^2 u_{i,j}}{\partial x^2} - \frac{h^3}{6}\frac{\partial^3 u_{i,j}}{\partial x^3} + O(h^4)\,.
\end{aligned}$$

Dufort-Frankel

The Dufort-Frankel algorithm is

$$\frac{u_{i,j+1}-u_{i,j-1}}{2k}=\frac{u_{i+1,j}-u_{i,j+1}-u_{i,j-1}+u_{i-1,j}}{h^2}\,.$$

Each term is now replaced from its Taylor series to obtain after some algebra

$$\frac{\partial u_{i,j}}{\partial t}-\frac{\partial^2 u_{i,j}}{\partial x^2}=-\Big(\frac{k}{h}\Big)^2\frac{\partial^2 u_{i,j}}{\partial t^2}+O(h^2)+O(k^2)+O(k^3/h^2)\,.$$

Any implementation of the Dufort-Frankel algorithm in which h and k individual tend to zero with $k/h=c$, a constant, will solve $u_t=u_{xx}-c^2u_{tt}$. Consequently the Dufort-Frankel algorithm is not consistent.

Crank-Nicolson

The Crank-Nicolson algorithm is

$$\frac{u_{i,j+1}-u_{i,j}}{k}=\frac{1}{2h^2}\Big[u_{i+1,j+1}+u_{i+1,j}-2(u_{i,j+1}+u_{i,j})+u_{i-1,j+1}+u_{i-1,j}\Big]\,.$$

As with the Dufort-Frankel algorithm, each term is now replaced from its Taylor series. The algebra is more complicated and so we list the individual components as follows:-

$$\begin{aligned}\frac{u_{i,j+1}-u_{i,j}}{k} &= \frac{\partial u_{i,j}}{\partial t}+\frac{k}{2}\frac{\partial^2 u_{i,j}}{\partial t^2}+O(k^2)\\ u_{i+1,j+1}-2u_{i,j+1}+u_{i-1,j+1} &= h^2\frac{\partial^2 u_{i,j+1}}{\partial x^2}+O(h^4)\\ u_{i+1,j}-2u_{i,j}+u_{i-1,j} &= h^2\frac{\partial^2 u_{i,j}}{\partial x^2}+O(h^4)\,.\end{aligned}$$

When these components are inserted into the Crank-Nicolson algorithm, the result is

$$\frac{\partial u_{i,j}}{\partial t}+\frac{k}{2}\frac{\partial^2 u_{i,j}}{\partial t^2}+O(k^2)=\frac{1}{2h^2}\Big[h^2\frac{\partial^2 u_{i,j+1}}{\partial x^2}+h^2\frac{\partial^2 u_{i,j}}{\partial x^2}+O(h^4)\Big]$$

which in turn simplifies to

$$\frac{\partial u_{i,j}}{\partial t}+\frac{k}{2}\frac{\partial^2 u_{i,j}}{\partial t^2}+O(k^2)=\frac{1}{2}\Big[2\frac{\partial^2 u_{i,j}}{\partial x^2}+k\frac{\partial^3 u_{i,j}}{\partial x^2\partial t}+O(k^2)+O(h^2)\Big]\,.$$

It is now straight forward algebra to observe that this equation can be rewritten in the form

$$\frac{\partial u_{i,j}}{\partial t}-\frac{\partial^2 u_{i,j}}{\partial x^2}=\frac{k}{2}\Big(\frac{\partial^3 u_{i,j}}{\partial x^2\partial t}-\frac{\partial^2 u_{i,j}}{\partial t^2}\Big)+O(k^2)+O(h^2)\,.$$

Thus the Crank-Nicolson algorithm is consistent as h and k tend to zero.

4.3 Numerical stability

Two methods, one including the effect of boundary conditions and the other excluding the effect of boundary conditions, are used to investigate stability. Both methods are attributed to John von Neumann. The approach which excludes the effect of boundary conditions is usually called the "Fourier method" while that which includes the effect of boundary conditions is usually called the "matrix method". In practice, it is normally assumed that the boundary conditions have a negligible impact on the stability of the numerical procedure.

4.3.1 Fourier method

The primary observation in the Fourier method is that the numerical scheme is linear and therefore it will have solutions in the form $u_{p,q} = \lambda^q\, e^{i\alpha ph}$. The numerical scheme is stable provided $|\lambda| < 1$ and unstable whenever $|\lambda| > 1$.

Euler method

The Euler method uses the numerical scheme

$$u_{p,q+1} = u_{p,q} + r(u_{p-1,q} - 2u_{p,q} + u_{p+1,q})$$

and therefore

$$\lambda^{q+1} e^{i\alpha ph} = \lambda^q e^{i\alpha ph} + r\Big(\, \lambda^q e^{i\alpha(p-1)h} - 2\lambda^q e^{i\alpha ph} + \lambda^q e^{i\alpha(p+1)h} \,\Big)\,.$$

This equation simplifies immediately to

$$\lambda = 1 + r\Big(\, e^{-i\alpha h} - 2 + e^{i\alpha h} \,\Big) = 1 + 2r\Big(\, \cos\alpha h - 1 \,\Big) = 1 - 4r\sin^2 \alpha h/2\,.$$

Clearly λ is real-valued and satisfies $\lambda < 1$. Therefore Euler's method is stable provided $\lambda > -1$ which in turn requires that

$$1 - 4r\sin^2 \alpha h/2 > -1 \quad \rightarrow \quad \frac{1}{2\sin^2 \alpha h/2} > r$$

for all choices of α. This condition can be satisfied only provided $r < 1/2$, and therefore the numerical stability of the Euler method requires that $r < 1/2$.

Richardson method

The Richardson method uses the numerical scheme

$$u_{p,q+1} = u_{p,q-1} + 2r(u_{p+1,q} - 2u_{p,q} + u_{p-1,q})$$

and therefore

$$\lambda^{q+1}e^{i\alpha ph} = \lambda^{q-1}e^{i\alpha ph} + 2r\Big(\,\lambda^q e^{i\alpha(p+1)h} - 2\lambda^q e^{i\alpha ph} + \lambda^q e^{i\alpha(p-1)h}\,\Big)\,.$$

This equation simplifies immediately to

$$\lambda = \lambda^{-1} + 2r\Big(\,e^{i\alpha h} - 2 + e^{-i\alpha h}\,\Big) = \lambda^{-1} + 4r\Big(\,\cos\alpha h - 1\,\Big) = \lambda^{-1} - 8r\sin^2\alpha h/2\,.$$

In conclusion, λ is the solution of the quadratic equation

$$\lambda^2 + 8\lambda r\sin^2\alpha h/2 - 1 = 0$$

with solutions

$$\lambda = -4r\sin^2\alpha h/2 \pm \sqrt{1 + 16r^2\sin^4\alpha h/2}\,.$$

Clearly both solutions are real but one solution satisfies $|\lambda| > 1$. Therefore Richardson's method is never stable (as was observed in the numerical example). The oscillatory nature of blow-up in Richardson's algorithm is due to the fact that instability ensues through a negative value of λ.

Dufort-Frankel method

The Dufort-Frankel method uses the numerical scheme

$$u_{p,q+1} = u_{p,q-1} + 2r(u_{p+1,q} - u_{p,q+1} - u_{p,q-1} + u_{p-1,q})$$

and therefore

$$\lambda^{q+1}e^{i\alpha ph} = \lambda^{q-1}e^{i\alpha ph} + 2r\Big(\,\lambda^q e^{i\alpha(p+1)h} - \lambda^{q+1}e^{i\alpha ph} - \lambda^{q-1}e^{i\alpha ph} + \lambda^q e^{i\alpha(p-1)h}\,\Big)\,.$$

This equation simplifies immediately to

$$\lambda = \lambda^{-1} + 2r(\,e^{i\alpha h} - \lambda - \lambda^{-1} + e^{-i\alpha h}\,) = -2r\lambda + (1-2r)\lambda^{-1} + 4r\cos(\alpha h)\,.$$

In conclusion, λ is the solution of the quadratic equation

$$(1+2r)\lambda^2 - 4\lambda r\cos\alpha h - (1-2r) = 0$$

with solutions

$$\lambda = \frac{2r\cos\alpha h \pm \sqrt{1 - 4r^2\sin^2\alpha h}}{1+2r}\,.$$

The roots of this quadratic are either both real or form a complex conjugate pair. If the roots are real, the triangle inequality gives

$$|\lambda| = \Big|\frac{2r\cos\alpha h \pm \sqrt{1-4r^2\sin^2\alpha h}}{1+2r}\Big| \le \frac{|\,2r\cos\alpha h\,| + \sqrt{1-4r^2\sin^2\alpha h}}{1+2r} \le \frac{2r+1}{1+2r} = 1\,.$$

On the other hand, if the roots are complex then $2r > 1$ and

$$|\lambda|^2 = \frac{4r^2\cos^2\alpha h + 4r^2\sin^2\alpha h - 1}{(1+2r)^2} = \frac{4r^2-1}{(1+2r)^2} \le \frac{2r-1}{2r+1} \le 1\,.$$

Either way, the Dufort-Frankel scheme is stable. This means that it will not blow-up due to the presence of rounding error, although it has already been shown to be inconsistent. It is now clear why this scheme converges, but not necessarily to the desired solution.

Crank-Nicolson method

The Crank-Nicolson method uses the numerical scheme

$$u_{p,q+1} = u_{p,q} + \frac{r}{2}\left[\, u_{p+1,q+1} + u_{p+1,q} - 2(u_{p,q+1} + u_{p,q}) + u_{p-1,q+1} + u_{p-1,q} \,\right]$$

and therefore

$$\lambda^{q+1}e^{i\alpha ph} = \lambda^q e^{i\alpha ph} + \frac{r}{2}\left[\, \lambda^{q+1}e^{i\alpha(p+1)h} + \lambda^q e^{i\alpha(p+1)h} \right.$$
$$\left. - 2\Big(\lambda^{q+1}e^{i\alpha ph} + \lambda^q e^{i\alpha ph}\Big) + \lambda^{q+1}e^{i\alpha(p-1)h} + \lambda^q e^{i\alpha(p-1)h} \,\right].$$

This equation simplifies immediately to

$$\lambda = 1 + \frac{r}{2}\left[\, \lambda e^{i\alpha h} + e^{i\alpha h} - 2(\lambda + 1) + \lambda e^{-i\alpha h} + e^{-i\alpha h} \,\right].$$

Further simplification gives

$$\lambda = 1 + r\,(\,\lambda\cos\alpha h + \cos\alpha h - \lambda - 1\,) \quad \rightarrow \quad \lambda = \frac{1 - 2r\sin^2\alpha h/2}{1 + 2r\sin^2\alpha h/2}\,.$$

Irrespective of the choice of k or h, it is clear that $|\lambda| < 1$ and therefore the Crank-Nicolson scheme is stable. This means that it will not blow-up due to the presence of rounding error.

4.3.2 Matrix method

The matrix method relies on Gershgorin's circle theorem regarding the location of matrix eigenvalues. The theorem asserts that if A is an $n \times n$ matrix and $\mathcal{C}_1, \cdots \mathcal{C}_n$ are the circles defined by

$$\mathcal{C}_k = \{X \,:\, |X - A_{kk}| \leq \sum_{j=1\; j\neq k}^{n} |A_{kj}|\,\}$$

then all the eigenvalues of A are contained within the region $\mathcal{D} = \mathcal{C}_1 U \mathcal{C}_2 \ldots U \mathcal{C}_n$. In particular, if m $(m \leq n)$ of these circles form a subspace of $\mathcal{D}$ which does not intersect the remaining $(n-m)$ circles, then precisely m eigenvalues are contained within such a subregion.

Example

Use Gerschgorin's theorem to locate the eigenvalues of the matrix.

$$A = \begin{bmatrix} 9 & 1 & 1 \\ 0 & 1 & 1 \\ -2 & 4 & 0 \end{bmatrix}$$

Solution

The Gershgorin circles are

$$\mathcal{C}_1 = \{Z \in \mathbb{C} : |Z - 9| = 2\}\,, \quad \mathcal{C}_2 = \{Z \in \mathbb{C} : |Z - 1| = 1\}\,, \quad \mathcal{C}_3 = \{Z \in \mathbb{C} : |Z| = 6\}\,.$$

Thus one eigenvalue lies in $|Z - 9| = 2$ and two eigenvalues lie in $|Z| = 6$.

Euler method

The Euler method is the algorithm

$$u_{p,q+1} = u_{p,q} + r(u_{p+1,q} - 2u_{p,q} + u_{p-1,q}) = (1-2r)u_{p,q} + ru_{p+1,q} + ru_{p-1,q}$$

where the values of $u_{0,q}$ and $u_{n,q}$ are given by the boundary conditions in the Dirichlet problem. The aim of the algorithm is to compute $u_{1,q}, \cdots, u_{n-1,q}$ for all values of q. Let $U^{(q)} = (u_{1,q}, \cdots, u_{n-1,q})^{\mathrm{T}}$ then Euler's algorithm corresponds to the matrix operation

$$U^{(q+1)} = TU^{(q)} + r\,F^{(q)} \tag{4.1}$$

where the $(n-1)\times(n-1)$ tri-diagonal matrix T and the $(n-1)$ dimensional vector $F^{(q)}$ are

$$T = \begin{bmatrix} 1-2r & r & 0 & 0 & \cdots & \cdots & 0 \\ r & 1-2r & r & 0 & \cdots & \cdots & 0 \\ 0 & r & 1-2r & r & 0 & \cdots & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \cdots \\ 0 & \cdots & \cdots & \cdots & 0 & r & 1-2r \end{bmatrix}, \qquad F^{(q)} = \begin{bmatrix} u_{0,q} \\ 0 \\ \vdots \\ 0 \\ u_{n,q} \end{bmatrix}.$$

Thus $U^{(1)} = TU^{(0)} + rF^{(0)}$ and

$$U^{(2)} = TU^{(1)} + rF^{(1)} = T(TU^{(0)} + rF^{(0)}) + rF^{(1)} = T^2U^{(0)} + rTF^{(0)} + rF^{(1)}\,.$$

These calculations suggest that equation (4.1) has general solution

$$U^{(q+1)} = T^{q+1}U^{(0)} + r\sum_{j=0}^{q} T^j F^{(q-j)}\,.$$

This result can be established by induction. Assuming that the boundary conditions are bounded functions of time, then the solution $U^{(q+1)}$ remains bound only provided $T^{q+1}U^{(0)}$ does not blow up. Since T is symmetric it has a complete set of (real) eigenvalues $\lambda_1, \cdots, \lambda_{n-1}$ with corresponding normalised eigenvectors $E_1, \cdots, E_{n-1}$. Let $U^{(0)} = c_1E_1 + \cdots + c_{n-1}E_{n-1}$ then

$$T^{q+1}U^{(0)} = \sum_{j=1}^{n-1} c_j\lambda_j^{q+1}E_j\,,$$

which will remain bounded as $t\to\infty$ only provided each eigenvalue of T lies within the unit circle. On the unit circle is not good enough in the presence of non-zero boundary information. Gershgorin's circle theorem asserts that all the eigenvalues of T (which are known to be real) lie within the region

$$\mathcal{D} = \{z\in\mathbb{C} : |z-(1-2r)| = r\} \cup \{z\in\mathbb{C} : |z-(1-2r)| = 2r\} = \{z\in\mathbb{C} : |z-(1-2r)| = 2r\}\,.$$

This is a circle centre $(1-2r, 0)$ and radius $2r$. The extremities of the circle on the x-axis are the points $(1-4r, 0)$ and $(1, 0)$. Stability is therefore guaranteed in the Euler algorithm provided $1-4r > -1$, that is, $r < 1/2$.

4.3.3 Gradient boundary conditions

Consider now the diffusion equation

$$u_t = u_{xx}\,, \qquad (t,x) \in (0,\infty)\times(0,1)$$

with initial condition $u(x,0) = f(x)$ and the gradient conditions $u_x(0,t) = u_x(1,t) = 0$. As previously, u_t is replaced by its forward difference approximation and u_{xx} is replaced by its central difference approximation to obtain

$$u_{i,j+1} = u_{i,j} + r\left(u_{i+1,j} - 2u_{i,j} + u_{i-1,j}\right) + O(k^2, kh^2)\,, \qquad i = 1,\cdots,(n-1)$$

in which r is the Courant number. The boundary conditions when expressed in terms of forward/backward differences of u give

$$\frac{-3u_{0,j} + 4u_{1,j} - u_{2,j}}{2h} = O(h^2)\,, \qquad \frac{3u_{n,j} - 4u_{n-1,j} + u_{n-2,j}}{2h} = O(h^2)\,.$$

The values of the solution at time $j+1$ can be computed from the solution at time j using the results

$$\begin{aligned}
u_{1,j+1} &= (1-2r)u_{1,j} + ru_{2,j} + ru_{0,j} + O(kh^2, k^2)\\
u_{2,j+1} &= (1-2r)u_{2,j} + ru_{3,j} + ru_{1,j} + O(kh^2, k^2)\\
u_{n-1,j+1} &= (1-2r)u_{n-1,j} + ru_{n,j} + ru_{n-2,j} + O(kh^2, k^2)\\
u_{n-2,j+1} &= (1-2r)u_{n-2,j} + ru_{n-1,j} + ru_{n-3,j} + O(kh^2, k^2)
\end{aligned}$$

These results are now used to compute $u_{0,j+1}$ and $u_{n,j+1}$ as follows:-

$$\begin{aligned}
u_{0,j+1} &= \frac{1}{3}\left[\,4u_{1,j+1} - u_{2,j+1}\,\right] + O(h^3)\\
&= \frac{1}{3}\left[\,(4-9r)u_{1,j} + (6r-1)u_{2,j} + 4ru_{0,j} - ru_{3,j}\,\right] + O(kh^2, k^2, h^3)\\
u_{n,j+1} &= \frac{1}{3}\left[\,4u_{n-1,j+1} - u_{n-2,j+1}\,\right] + O(h^3)\\
&= \frac{1}{3}\left[\,(4-9r)u_{n-1,j} + 4ru_{n,j} + (6r-1)u_{n-2,j} - ru_{n-3,j}\,\right] + O(kh^2, k^2, h^3)
\end{aligned}$$

Now let $U^{(j)} = (u_{0,j},\cdots,u_{n,j})^{\mathrm{T}}$ then the Euler scheme for the determination of the solution becomes

$$U^{(j+1)} = AU^{(j)}$$

where A is the $(n+1)\times(n+1)$ matrix

$$\begin{bmatrix} \frac{4r}{3} & \frac{4-9r}{3} & \frac{6r-1}{3} & -\frac{r}{3} & 0 & \cdots & 0 \\ r & 1-2r & r & 0 & \cdots & \cdots & 0 \\ 0 & r & 1-2r & r & 0 & \cdots & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & b \\ 0 & \cdots & \cdots & -\frac{r}{3} & \frac{6r-1}{3} & \frac{4-9r}{3} & \frac{4r}{3} \end{bmatrix}$$

Clearly A is neither symmetric or tri-diagonal. Nevertheless Gershgorin's circle theorem indicates that all the eigenvalues of A are located within the region

$$\mathcal{D} = \{z \in \mathbb{C} : |z-(1-2r)| = 2r\} \cup \{z \in \mathbb{C} : |3z-4r| = r + |4-9r| + |6r-1|\}.$$

As previously, the circle centre (1-2r,0) lies within the unit circle provided $r < 1/2$. However the analysis of the second circle requires the treatment of several cases. The boundary values are $r = 4/9$ and $r = 1/6$.

Case 1 Suppose $0 < r < 1/6$ then $0 < 9r < 3/2 < 4$. The second circle is

$$|3z-4r| = r + 4 - 9r + 1 - 6r = 5 - 14r \quad \rightarrow \quad |3z-4r| = 5-14r\,.$$

The line segment of the x-axis lying within this circle is $-5+14r \le 3x-4r \le 5-14r$, which becomes after simplification $-5/3+6r \le x \le 5/3-10r/3$. This line segment lies within the unit circle provided $5/3-10r/3 < 1$ **and** $-1 < -5/3+6r$, that is, provided $1/5 < r$ **and** $1/9 < r$. But $r < 1/6$ by assumption and so these conditions cannot be satisfied.

Case 2 When $1/6 < r < 4/9$, the second circle has equation

$$|3z-4r| = r + 4 - 9r + 6r - 1 = 3 - 2r \quad \rightarrow \quad |3z-4r| = 3-2r\,.$$

The line segment of the x-axis lying within this circle is $-3+2r \le 3x-4r \le 3-2r$, which becomes after simplification $-1+2r \le x \le 1+2r/3$. Clearly $1+2r/3 > 1$ for $r > 0$ and so this line segment never lies within the unit circle.

Case 3 When $4/9 < r < 1/2$, the second circle has equation

$$|3z-4r| = r + 9r - 4 + 6r - 1 = 16r - 5 \quad \rightarrow \quad |3z-4r| = 16r-5\,.$$

The line segment of the x-axis lying within this circle is $-16r+5 \le 3x-4r \le 16r-5$, which becomes after simplification $5/3-4r \le x \le 20r/3-5/3$. This line segment lies within the unit circle provided

$20r/3 - 5/3 < 1$ **and** $-1 < 5/3 - 4r$, that is, provided $r < 2/5$ **and** $r < 2/3$. But $r > 4/9$ and so again these conditions cannot be satisfied.

In conclusion, Gershgorin's theorem on this occasion provides no useful information regarding the stability of the algorithm. In fact, the issue of stability is of secondary importance in this problem to the issue of accuracy. The detailed analysis of the error structure in this Neumann problem indicates that the numerical scheme no longer has the expected $O(kh^2, k^2)$ accuracy of the Dirichlet problem. The error at each iteration is dominated by $O(h^3)$ which behaves like $O(kh)$ assuming that $k = rh^2 = O(1)$. the treatment of the Neumann boundary value problem for the diffusion equation is non-trivial and is not pursued further here.

4.4 Numerical convergence - The Lax equivalence theorem

Given a properly posed linear initial boundary value problem and a linear finite difference approximation to it that is consistent and stable, then the scheme is convergent.

The problem is properly posed if

(i) The solution is unique when it exists;

(ii) The solution depends continuously on the initial data;

(iii) A solution always exists for initial data arbitrarily close to initial data for which a solution does not exist.

Example Let $e_{i,j} = u_{i,j} - \widehat{u}(x_i, t_j)$ where $\widehat{u}(x,t)$ is the exact solution at (x,t). Euler's algorithm

$$u_{i,j+1} = u_{i,j} + r(u_{i+1,j} - 2u_{i,j} + u_{i-1,j})$$

can be re-expressed in the form

$$\widehat{u}_{i,j+1} + e_{i,j+1} = \widehat{u}_{i,j} + e_{i,j} + r(\widehat{u}_{i+1,j} - 2\widehat{u}_{i,j} + \widehat{u}_{i-1,j}) + r(e_{i+1,j} - 2e_{i,j} + e_{i-1,j})\,.$$

Taylors theorem may now be use to assert that

$$\begin{aligned}
\widehat{u}_{i,j+1} &= \widehat{u} + k\,\widehat{u}_t + \frac{k^2}{2}\,\widehat{u}_{tt} + O(k^3) \\
\widehat{u}_{i+1,j} &= \widehat{u} + h\,\widehat{u}_x + \frac{h^2}{2}\,\widehat{u}_{xx} + \frac{h^3}{6}\,\widehat{u}_{xxx} + \frac{h^4}{24}\,\widehat{u}_{xxxx} + O(h^5) \\
\widehat{u}_{i-1,j} &= \widehat{u} - h\,\widehat{u}_x + \frac{h^2}{2}\,\widehat{u}_{xx} - \frac{h^3}{6}\,\widehat{u}_{xxx} + \frac{h^4}{24}\,\widehat{u}_{xxxx} + O(h^5)
\end{aligned}$$

where $\widehat{u} = \widehat{u}(x,t)$. These formulae are now inserted into the numerical scheme to obtain

$$\begin{aligned}
k\,\widehat{u}_t + \frac{k^2}{2}\,\widehat{u}_{tt} + O(k^3) + e_{i,j+1} &= e_{i,j} + r\Big[\,h^2\,\widehat{u}_{xx} + \frac{h^4}{12}\,\widehat{u}_{xxxx} + O(h^6)\,\Big] \\
&\quad + r(e_{i+1,j} - 2e_{i,j} + e_{i-1,j})\,.
\end{aligned}$$

after all elementary simplifications have been done. The courant number in the second term on the right hand side of this equation is now replaced by its definition to get

$$\begin{aligned} e_{i,j+1} &= (1-2r)e_{i,j} + k\,(\widehat{u}_{xx} - \widehat{u}_t) + \frac{kh^2}{12}\,\widehat{u}_{xxxx} - \frac{k^2}{2}\,\widehat{u}_{tt} + O(k^3, kh^4) \\ &\quad + r\,e_{i+1,j} + r\,e_{i-1,j}\,. \end{aligned}$$

Since $\widehat{u}$ satisfies $\widehat{u}_t = \widehat{u}_{xx}$ then it also follows that $\widehat{u}_{tt} = \widehat{u}_{xxxx}$. With these further simplifications, it is clear that

$$e_{i,j+1} = (1-2r)e_{i,j} + \frac{kh^2}{12}(1-6r)\,\widehat{u}_{tt} + r\,e_{i+1,j} + r\,e_{i-1,j} + O(k^3, kh^4)\,.$$

Now define E_j to be the maximum spatial discretisation error at time t_j and let M be the maximum value of $\widehat{u}_{tt}$ across space and time then the triangle inequality yields initially

$$|\,e_{i,j+1}\,| \le |\,1-2r\,|\,|\,e_{i,j}\,| + \frac{kh^2}{12}\,|1-6r\,|\,M + r\,|\,e_{i+1,j}\,| + r\,|\,e_{i-1,j}\,| + O(k^3, kh^4)\,,$$

which in turn indicates that E_j satisfies

$$E_{j+1} \le (2r + |\,1-2r\,|\,)\,E_j + \frac{kh^2}{12}\,|1-6r\,|\,M + O(k^3, kh^4)\,.$$

Assuming that $r \le 1/2$, then $|\,1-2r\,| = 1-2r$ and it now obvious that

$$E_{j+1} \le E_j + \frac{kh^2}{12}\,|1-6r\,|\,M + O(k^3, kh^4)$$

with solution

$$E_j \le E_0 + \frac{jkh^2}{12}\,|1-6r\,|\,M + jkO(k^2, h^3)\,.$$

Since $jk = T$, the current time interval covered by the calculation, it follows immediately that

$$E(T) \le E(0) + T\left[\,\frac{Mh^2|1-6r\,|}{12} + O(k^2, h^4)\right].$$

Thus provided $r < 1/2$ then the maximum spatial discretisation error in the Euler scheme grows in direct proportion to T. In particular, the Euler scheme is $O(k)$ more accurate than might otherwise have been anticipated when $r = 1/6$.

Exercises

Question 27.

The equation $u_t = u_{xx}$ is approximated at the point (ih, jk) by

$$\theta\left(\frac{u_{i,j+1} - u_{i,j-1}}{2k}\right) + (1-\theta)\left(\frac{u_{i,j} - u_{i,j-1}}{k}\right) - \left(\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2}\right) = 0,$$

Show that the truncation error is given by

$$-\frac{k(1-\theta)}{2}\frac{\partial^2 u}{\partial t^2} - \frac{h^2}{12}\frac{\partial^4 u}{\partial x^4} + O(k^2, h^4).$$

What is the optimal value for θ?

Question 28.

The function $u(x,t)$ satisfies the equation

$$u_t = u_{xx}, \qquad (x,t) \in (0,1) \times (0,\infty)$$

with initial condition $u(x,0) = \sin \pi x$ and boundary conditions $u(0,t) = u(1,t) = 0$ for $t > 0$.

Write out the discretised scheme using the explicit Euler method with a step length $h = 1/4$ and a time step $k = 0.05$. Use the symmetry of the problem to reduce the number of unknowns. Solve the equations to find an approximate value for $u(1/2, 0.1)$. [You could compare your numerical solution with the exact one obtained via separation of variables].

Question 29.

Derive the Crank-Nicolson equations for the previous problem. Use these with $h = 1/4$, $k = 0.1$ to estimate the value of $u(1/2, 0.1)$ [exact answer 0.3727]. Also, with $h = 1/4$ and $k = 0.025$ estimate $u(0.5, 0.05)$.

Question 30.

The function $u(x,t)$ satisfies the equation

$$u_t = xu_{xx}, \qquad (x,t) \in (0,1/2) \times (0,\infty)$$

with initial condition $u(x,0) = x(1-x)$ and boundary conditions $u(0,t) = 0$ and $\partial u(1/2,t)/\partial x = -u(1/2,t)/2$.

Write out the discretised scheme using the explicit Euler method with a step length $h = 0.1$ and a time step $k = 0.005$ using the fictitious point approach to the derivative boundary condition. [No need to solve.] What will happen around $(1/2, 0)$? Why?

Question 31.

Use Gershgorin Circles to prove that the Euler method used in the previous question with a general $(N+1)$-node mesh will not grow exponentially provided the Courant number r satisfies

$$r \leq \frac{4}{4+h}.$$

Question 32.

The equation $\alpha u_t + u_x - f(x,t) = 0$ in which α is a constant is approximated at the point (ih, jk) by

$$\frac{\alpha}{2k}\Big[\, 2u_{i,j+1} - (u_{i+1,j} + u_{i-1,j}) \,\Big] + \frac{(u_{i+1,j} - u_{i-1,j})}{2h} - f_{i,j} = 0.$$

Investigate the consistency of this method for $r = k/h$ and for $r = k/h^2$. If either is inconsistent, what equation does the method solve?

Question 33.

The equation $u_t = u_{xx}$ with $(x,t) \in (0,1)\times(0,\infty)$ is approximated at the point (ih, jk) by the scheme

$$u_{i,j+1} - u_{i,j} = r\left[\, \theta(u_{i-1,j+1} - 2u_{i,j+1} + u_{i+1,j+1}) + (1-\theta)(u_{i-1,j} - 2u_{i,j} + u_{i+1,j}) \,\right]$$

where $\theta \in [0,1]$, $r = k/h^2$ is the Courant number and $h = 1/N$. Assuming that the boundary conditions and initial conditions are known, investigate the Fourier stability of this method.

Question 34.

Show that the $n \times n$ tri-diagonal matrix

$$\begin{bmatrix} a & b & 0 & 0 & \cdots & \cdots & 0 \\ c & a & b & 0 & \cdots & \cdots & 0 \\ 0 & c & a & b & 0 & \cdots & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & b \\ 0 & \cdots & \cdots & \cdots & 0 & c & a \end{bmatrix}$$

has eigenvalues

$$\lambda_k = a + 2\sqrt{bc}\,\cos\theta_k\,, \qquad k = 1, \cdots, n$$

with corresponding eigenvector

$$X_k = \Big[\Big(\frac{c}{b}\Big)^{1/2}\sin\theta_k, \Big(\frac{c}{b}\Big)\sin 2\theta_k, \cdots, \Big(\frac{c}{b}\Big)^{j/2}\sin j\theta_k, \cdots, \Big(\frac{c}{b}\Big)^{n/2}\sin n\theta_k\,\Big]^{\mathrm{T}}.$$

where $\theta_k = k\pi/(n+1)$.

Question 35.

Let α be a constant. The equation $u_t = \alpha u_{xx}$ is to be solved in the region $(x,t) \in (0,1) \times (0,\infty)$ with boundary conditions $u(0,t) = u(1,t) = 0$. The solution is approximated at the point (ih, jk) by the fully implicit backward Euler scheme

$$u_{i,j+1} - u_{i,j} = r\alpha(u_{i-1,j+1} - 2u_{i,j+1} + u_{i+1,j+1}),$$

where $r = k/h^2$ and $h = 1/N$. Assuming that the initial conditions are known, investigate the stability of this method using both the matrix and Fourier methods.

[Recall that the eigenvalues of a constant $n \times n$ tri-diagonal matrix are

$$\lambda_i = a + 2\sqrt{bc}\cos\frac{i\pi}{n+1}$$

where a is the diagonal entry, b, and c are the sub- and super=diagonal entries.]

Question 36.

Show that the Euler and Richardson methods are both consistent with the original equation $u_t = u_{xx}$. So their difficulties are entirely due to their instabilities.

Chapter 5

Parabolic equations in two dimensions

5.1 Introduction

The canonical parabolic equation in two dimensions is

$$u_t = u_{xx} + u_{yy}\,, \qquad (x,y) \in \mathcal{D} \quad t \in (0,\infty)$$

with initial condition $u(x,y,0) = u_0(x,y)$ and boundary conditions at along $\partial\mathcal{D}$ which may be either Dirichlet (*i.e.* u), Neumann (*i.e.* $\partial u/\partial n$) or Robin (*i.e.* $\partial u/\partial n + \kappa u = 0$). As with elliptic equations in two dimensions, it will be assumed *a priori* that the spatial discretisation on the x and y directions is h. The finite difference formulation of $u_t = u_{xx} + u_{yy}$ is

$$\frac{du_{i,j}}{dt} = \frac{1}{h^2}\left[\left(u_{i+1,j} - 2u_{i,j} + u_{i-1,j}\right) + \left(u_{i,j+1} - 2u_{i,j} + u_{i,j-1}\right)\right]$$

The Crank-Nicholson re-formulation of this equation is obtained by integrating the equation over $[t_q, t_{q+1}]$ and approximating each integral on the right hand side by the trapezoidal rule to obtain

$$\begin{aligned} u_{i,j}^{q+1} &= u_{i,j}^{q} + \frac{r}{2}\left[\left(u_{i+1,j}^{q+1} - 2u_{i,j}^{q+1} + u_{i-1,j}^{q+1}\right) + \left(u_{i,j+1}^{q+1} - 2u_{i,j}^{q+1} + u_{i,j-1}^{q+1}\right)\right. \\ &\qquad \left. + \left(u_{i+1,j}^{q} - 2u_{i,j}^{q} + u_{i-1,j}^{q}\right) + \left(u_{i,j+1}^{q} - 2u_{i,j}^{q} + u_{i,j-1}^{q}\right)\right] \end{aligned}$$

which is subsequently re-arranged to give

$$\begin{aligned} u_{i,j}^{q+1} - \frac{r}{2}\left(u_{i+1,j}^{q+1} - 2u_{i,j}^{q+1} + u_{i-1,j}^{q+1}\right) - \frac{r}{2}\left(u_{i,j+1}^{q+1} - 2u_{i,j}^{q+1} + u_{i,j-1}^{q+1}\right) \\ = u_{i,j}^{q} + \frac{r}{2}\left(u_{i+1,j}^{q} - 2u_{i,j}^{q} + u_{i-1,j}^{q}\right) + \frac{r}{2}\left(u_{i,j+1}^{q} - 2u_{i,j}^{q} + u_{i,j-1}^{q}\right). \end{aligned}$$

As with the Crank-Nicolson formulation of the one dimensional problem, it may be demonstrated that this scheme is consistent. However, the matrix of the system is no longer tri-diagonal and so an iterative procedure is required to advance the solution through each time step. To simplify the representation of the scheme, the finite difference operator δ is introduce through the definition

$$\delta(x_p) = x_{p+1/2} - x_{p-1/2}\,.$$

Thus

$$\delta^2(x_p) = \delta(x_{p+1/2} - x_{p-1/2}) = (x_{p+1} - x_p) - (x_p - x_{p-1}) = x_{p+1} - 2x_p + x_{p-1}\,.$$

With this definition, the Crank-Nicolson scheme becomes

$$u_{i,j}^{q+1} - \frac{r}{2}\,\delta_x^2\,(\,u_{i,j}^{q+1}\,) - \frac{r}{2}\,\delta_y^2\,(\,u_{i,j}^{q+1}\,) = u_{i,j}^{q} + \frac{r}{2}\,\delta_x^2\,(\,u_{i,j}^{q}\,) + \frac{r}{2}\,\delta_y^2\,(\,u_{i,j}^{q}\,)\,.$$

5.2 Alternating Direction Implicit

The construction of the alternating direction implicit (ADI) scheme begins with the finite difference scheme

$$u_{i,j}^{q+1} - u_{i,j}^{q} = r\left[\,\delta_x^2\,(\,u_{i,j}^{q}\,) + \delta_y^2\,(u_{i,j}^{q}\,)\right]$$

in which the time derivative in the original equation has been replaced by its forward difference. A further time step gives

$$u_{i,j}^{q+2} - u_{i,j}^{q+1} = r\left[\,\delta_x^2\,(\,u_{i,j}^{q+1}\,) + \delta_y^2\,(u_{i,j}^{q+1}\,)\right].$$

The term $\delta_x^2\,(\,u_{i,j}^{q}\,)$ in the first of these equations is replaced by $\delta_x^2\,(\,u_{i,j}^{q+1}\,)$ and the term $\delta_y^2\,(\,u_{i,j}^{q+1}\,)$ in the second of these equations is replaced by $\delta_y^2\,(\,u_{i,j}^{q+2}\,)$. This procedure gives the two-step algorithm

$$\begin{aligned} u_{i,j}^{q+1} - r\,\delta_x^2\,(\,u_{i,j}^{q+1}\,) &= u_{i,j}^{q} + r\,\delta_y^2\,(u_{i,j}^{q}\,) \\ u_{i,j}^{q+2} - r\,\delta_y^2\,(\,u_{i,j}^{q+2}\,) &= u_{i,j}^{q+1} + r\,\delta_x^2\,(u_{i,j}^{q+1}\,)\,. \end{aligned}$$

By this methodology, $\delta_y^2\,(u_{i,j}^{q}\,)$ is explicit and $u_{i,j}^{q+1}$ is determined implicitly. The roles are reversed in the second equation. Here $\delta_x^2\,(u_{i,j}^{q+1}\,)$ is explicit and $u_{i,j}^{q+2}$ is determined implicitly. This is why origin of the "alternating direction" description of the algorithm. The important property of the ADI algorithm is that both steps only require the solution of tri-diagonal systems of linear equations, the first to find $u_{i,j}^{q+1}$ and the second to find $u_{i,j}^{q+2}$.

5.2.1 Consistency of ADI

For representational convenience, the subscripts are dropped from u. The ADI algorithm is

$$\begin{aligned} (1 - r\,\delta_x^2\,)u^{q+1} &= (1 + r\,\delta_y^2\,)u^{q} \\ (1 - r\,\delta_y^2\,)u^{q+2} &= (1 + r\,\delta_x^2\,)u^{q+1} \end{aligned}$$

which becomes on elimination of the terms at t_{q+1}

$$(1 - r\,\delta_x^2\,)(1 - r\,\delta_y^2\,)u^{q+2} = (1 + r\,\delta_x^2\,)(1 + r\,\delta_y^2\,)u^{q}\,.$$

It is now straight forward algebra to verify that

$$(1 + r^2\,\delta_x^2\,\delta_y^2\,)(u^{q+2} - u^{q}) = r\,(\delta_x^2 + \delta_y^2\,)(u^{q+2} + u^{q})\,.$$

This equation is now divided by $2k$ and r replaced by its definition to obtain

$$(1 + r^2\,\delta_x^2\,\delta_y^2\,)\frac{u^{q+2} - u^q}{2k} = (\delta_x^2 + \delta_y^2\,)\frac{u^{q+2} + u^q}{2h^2}\,.$$

Each term appearing in this equation is now replaced by its Taylor series expansion taken about (x_i, y_j) at time t_{q+1}. It is easy to see that $(u^{q+2} - u^q)/2k$ is the central difference formula for u_t at time t_{q+1} and therefore

$$\frac{u^{q+2} - u^q}{2k} = u_t^{q+1} + O(k^2)\,, \qquad u^{q+2} + u^q = 2u^{q+1} + O(k^2)\,.$$

Consequently,

$$(\delta_x^2 + \delta_y^2\,)\frac{u^{q+2} + u^q}{2h^2} = \frac{(\delta_x^2 + \delta_y^2)2u_t^{q+1}}{2h^2} + O(k^2) = u_{xx}^{q+1} + u_{yy}^{q+1} + O(k^2, h^2)\,.$$

Finally,

$$\frac{r^2\,\delta_x^2\,\delta_y^2\,(u^{q+2} - u^q)}{2k} = \frac{k^2}{h^4}\,\delta_x^2\,\delta_y^2\,(u_t^{q+1}) + O(k^2) = k^2\,u_{txxyy}^{q+1} + O(k^2, h^2)\,.$$

The component parts are now assembled to give

$$u_t^{q+1} + O(k^2) + k^2\,u_{txxyy}^{q+1} + O(r^2k^2) = u_{xx}^{q+1} + u_{yy}^{q+1} + O(k^2, h^2)$$

which may be rearranged to give

$$u_t^{q+1} - u_{xx}^{q+1} - u_{yy}^{q+1} = O(k^2, h^2)\,.$$

Consequently the ADI algorithm is unconditionally consistent.

5.2.2 Stability of the ADI algorithm

The stability of the ADI algorithm is investigated by the substitution

$$u_{p,q}^n = \lambda^n e^{i(\alpha p + \beta q)h}\,.$$

The calculation is based on the auxiliary observation that

$$\begin{aligned}\delta_x^2\,u &= (e^{i\alpha h} - 2 + e^{-i\alpha h})\,u = 2(\cos\alpha h - 1)\,u = -4(\sin^2\alpha h/2)\,u\,,\\ \delta_y^2\,u &= (e^{i\beta h} - 2 + e^{-i\beta h})\,u = 2(\cos\beta h - 1)\,u = -4(\sin^2\beta h/2)\,u\,.\end{aligned}$$

These formulae are now substituted into the numerical scheme

$$(1 - r\,\delta_x^2\,)(1 - r\,\delta_y^2\,)u^{q+2} = (1 + r\,\delta_x^2\,)(1 + r\,\delta_y^2\,)u^q\,.$$

to obtain

$$\lambda^2 = \frac{(1 - 4r\sin^2(\alpha h/2)\,)(1 - 4r\sin^2(\beta h/2)\,)}{(1 + 4r\sin^2(\alpha h/2)\,)(1 + 4r\sin^2(\beta h/2)\,)}\,.$$

The triangle inequality applied to the expression for λ^2 yields immediately that $|\lambda|^2 < 1$, and so the ADI algorithm is unconditionally stable for all choices of r.

Exercises

Question 37.

Show that the 2-D ADI method

$$\begin{aligned}(1 - r\delta_x^2)\bar{u}^{n+1} &= (1 + r\delta_y^2)\, u^n,\\ (1 - r\delta_y^2)u^{n+1} &= \bar{u}^{n+1} - r\delta_y^2 u^n\end{aligned}$$

is unconditionally stable. [Start by eliminating $\bar{u}^{n+1}$.]

Question 38.

Show that the 3-D ADI method

$$\begin{aligned}(1 - r\delta_x^2)u^{n+1} &= (1 + r\delta_y^2 + r\delta_z^2)u^n,\\ (1 - r\delta_y^2)u^{n+2} &= (1 + r\delta_x^2 + r\delta_z^2)u^{n+1},\\ (1 - r\delta_z^2)u^{n+3} &= (1 + r\delta_x^2 + r\delta_y^2)u^{n+2}\end{aligned}$$

is stable provided that $r \leq 1/2$. [Start by eliminating u^{n+1} and u^{n+2}.]

Question 39.

The function $u(x,t)$ satisfies the equation

$$u_t = u_{xx} + f(x,t)\,, \qquad (x,t) \in (0,1) \times (0,\infty)$$

with given initial condition and boundary conditions

$$u(0,t) = 0\,, \qquad \frac{\partial u(1,t)}{\partial x} = g(t)\,.$$

Show by careful analysis of truncation error that the gradient boundary condition derived by the fictitious nodes procedure usually gives rise to a system of finite difference equations that are $O(h)$ accurate.

Show that $O(h^3)$ accuracy is recovered provided

$$\frac{dg}{dt} = \frac{\partial f(x_n,t)}{\partial x}\,.$$

Chapter 6

NSPDE solutions

Solution 1.

(a) Define $g(h) = 2hf'(x) - f(x+h) + f(x-h)$ then clearly $g(0) = 0$. Three differentiations with respect to h give

$$\begin{aligned} \frac{dg}{dh} &= 2f'(x) - f'(x+h) - f'(x-h) \\ \frac{d^2g}{dh^2} &= -f''(x+h) + f''(x-h) \\ \frac{d^3g}{dh^3} &= -f'''(x+h) - f'''(x-h)\,. \end{aligned}$$

Since $g'(0) = g''(0) = 0$, then Maclaurin's theorem applied to $g(h)$ gives

$$g(h) = g(0) + hg'(0) + \frac{h^2}{2}\,g''(0) + \frac{h^3}{6}\,g'''(\lambda h) = \frac{h^3}{6}\,g'''(\lambda h)\,, \qquad \lambda \in (0,1)\,.$$

Given any continuous function $y(x)$, the task is now to show that the equation $q(\theta) = y(x+h) + y(x-h) - 2y(x+\theta h)$ has a solution in the interval $[-1,1]$. To establish this result, it is enough to observe that $q(-1) = y(x+h) - y(x-h)$ and $q(1) = y(x-h) - y(x+h)$. Clearly $q(1) + q(-1) = 0$ and therefore $q(\lambda)$ has at least one zero in $[-1,1]$. Consequently there exist η such that

$$\frac{d^3g}{dh^3} = -f'''(x+h) - f'''(x-h) = -2f'''(\eta)\,, \qquad \eta \in [x-h, x+h]$$

and the result is proves by asserting that

$$g(h) = 2hf'(x) - f(x+h) + f(x-h) = -\frac{h^3}{3}\,f'''(\eta)\,.$$

which in turn simplifies to the answer

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} - \frac{h^2}{6}\,f'''(\eta)\,.$$

(b) Define $g(h) = h^2 f''(x) - f(x+h) + 2f(x) - f(x-h)$ then clearly $g(0) = 0$. Four differentiations with respect to h give

$$\begin{aligned}
\frac{dg}{dh} &= 2hf''(x) - f'(x+h) + f'(x-h) \\
\frac{d^2g}{dh^2} &= 2f''(x) - f''(x+h) - f''(x-h) \\
\frac{d^3g}{dh^3} &= -f'''(x+h) + f'''(x-h) \\
\frac{d^4g}{dh^4} &= -f''''(x+h) - f''''(x-h)\,.
\end{aligned}$$

Since $g'(0) = g''(0) = g'''(0) = 0$, then Maclaurin's theorem applied to $g(h)$ gives

$$g(h) = g(0) + hg'(0) + \frac{h^2}{2}\,g''(0) + \frac{h^3}{6}\,g'''(0) + \frac{h^4}{24}\,g''''(\lambda h) = \frac{h^4}{24}\,g''''(\lambda h)\,, \qquad \lambda \in (0,1)\,.$$

Taking $y(x) = f''''(x)$ in part (a), it follows that there exist η such that

$$\frac{d^4g}{dh^4} = -f''''(x+h) - f''''(x-h) = -2f''''(\xi)\,, \qquad \xi \in [x-h, x+h]$$

and the result is proved by asserting that

$$g(h) = h^2 f'(x) - f(x+h) + 2f(x) - f(x-h) = -\frac{h^4}{12}\,f''''(\xi)\,.$$

which in turn simplifies to the answer

$$f''(x) = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} - \frac{h^2}{12}\,f''''(\xi)\,.$$

Solution 2.

The Taylor series expansion of $f(x+h)$ is

$$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2}f''(x) + \frac{h^3}{6}f'''(x) + O(h^4)\,.$$

The Taylor series of $4f(x+h) - f(x+2h) - 3f(x)$ is therefore

$$\begin{aligned}
& 4\Big[f(x) + hf'(x) + \frac{h^2}{2}f''(x) + \frac{h^3}{6}f'''(x) + O(h^4)\Big] \\
& \quad - \Big[f(x) + 2hf'(x) + \frac{(2h)^2}{2}f''(x) + \frac{(2h)^3}{6}f'''(x) + O(h^4)\Big] - 3f(x) \\
= \;& 4f(x) + 4hf'(x) + \frac{4h^2}{2}f''(x) + \frac{4h^3}{6}f'''(x) \\
& \quad - f(x) - 2hf'(x) - \frac{4h^2}{2}f''(x) - \frac{8h^3}{6}f'''(x) - 3f(x) + O(h^4) \\
= \;& 2hf'(x) - \frac{2h^3}{3}f'''(x) + O(h^4)\,.
\end{aligned}$$

It now follows by straight forward algebra that

$$\begin{aligned} f'(x) &= \frac{4f(x+h) - f(x+2h) - 3f(x)}{2h} + \frac{h^2}{3}f'''(x) + O(h^4) \\ &= \frac{4f(x+h) - f(x+2h) - 3f(x)}{2h} + O(h^2)\,. \end{aligned}$$

This is the *forward difference formula* for $f'(x)$. A *backward difference formula* can be obtained by replacing h by $-h$ to obtain

$$\begin{aligned} f'(x) &= \frac{4f(x-h) - f(x-2h) - 3f(x)}{-2h} + \frac{h^2}{3}f'''(x) + O(h^4) \\ &= \frac{f(x-2h) - 4f(x-h) + 3f(x)}{2h} + O(h^2)\,. \end{aligned}$$

Solution 3.

These results can be established directly using Taylor series in an efficient way by recognising that the finite difference expressions involve very specific combinations of function values. Define

$$\psi(x,h) = u(x+h) - u(x-h)\,, \qquad \phi(x,h) = u(x+h) + u(x-h)\,.$$

then Taylor's theorem gives immediately that

$$\begin{aligned} \psi(x,h) &= 2hu^{(1)}(x) + \frac{h^3}{3}u^{(3)}(x) + \frac{h^5}{60}u^{(5)}(x) + O(h^7) \\ \phi(x,h) &= 2u(x) + h^2u^{(2)}(x) + \frac{h^4}{12}u^{(4)}(x) + O(h^6)\,. \end{aligned}$$

(a) It now follows by straight forward algebra that

$$\begin{aligned} u(x+2h) - 2u(x+h) + 2u(x-h) - u(x-2h) &= \psi(x,2h) - 2\psi(x,h) \\ &= \Big[4hu^{(1)}(x) + \frac{8h^3}{3}u^{(3)}(x)\Big] - 2\Big[2hu^{(1)}(x) + \frac{h^3}{3}u^{(3)}(x)\Big] + O(h^5) \\ &= 2h^3u^{(3)}(x) + O(h^5)\,. \end{aligned}$$

This result and the definition of "Big O" indicate that

$$u^{(3)}(x) = \frac{u(x+2h) - 2u(x+h) + 2u(x-h) - u(x-2h)}{2h^3} + O(h^2)\,.$$

(b) By a similar piece of straight forward algebra gives

$$\begin{aligned} u(x+2h) - 4u(x+h) + 6u(x) - 4u(x-h) + u(x-2h) &= \phi(x,2h) - 4\phi(x,h) + 6u(x) \\ &= \Big[2u(x) + 4h^2u^{(2)}(x) + \frac{16h^4}{12}u^{(4)}(x)\Big] - 4\Big[2u(x) + h^2u^{(2)}(x) + \frac{h^4}{12}u^{(4)}(x)\Big] + 6u(x) + O(h^6) \\ &= h^4u^{(4)}(x) + O(h^6)\,. \end{aligned}$$

This result and the definition of "Big O" indicate that

$$u^{(4)}(x) = \frac{u(x+2h) - 4u(x+h) + 6u(x) - 4u(x-h) + u(x-2h)}{h^4} + O(h^2)\,.$$

Solution 4.

These results are established directly using the Taylor series expansions

$$\begin{aligned}\psi(x,h) &= u(x+h) - u(x-h) = 2hu^{(1)}(x) + \frac{h^3}{3}u^{(3)}(x) + \frac{h^5}{60}u^{(5)}(x) + O(h^7)\\ \phi(x,h) &= u(x+h) + u(x-h) = 2u(x) + h^2u^{(2)}(x) + \frac{h^4}{12}u^{(4)}(x) + O(h^6)\,.\end{aligned}$$

(a) It follows by straight forward algebra that

$$\begin{aligned}u(x-2h) + 8u(x-h) - 8u(x+h) - u(x+2h) &= 8\psi(x,h) - \psi(x,2h)\\ &= 8\Big[2hu^{(1)}(x) + \frac{h^3}{3}u^{(3)}(x)\Big] - \Big[4hu^{(1)}(x) + \frac{8h^3}{3}u^{(3)}(x)\Big] + O(h^5)\\ &= 12hu^{(1)}(x) + O(h^5)\,.\end{aligned}$$

This result and the definition of "Big O" indicate that

$$u^{(1)}(x) = \frac{u(x-2h) - 8u(x-h) + 8u(x+h) - u(x+2h)}{12h} + O(h^4)\,.$$

(b) It follows by straight forward algebra that

$$\begin{aligned}-u(x-2h) + 16u(x-h) - 30u(x) + 16u(x+h) - u(x-2h) &= 16\phi(x,h) - \phi(x,2h) - 30u(x)\\ = 16\Big[2u(x) + h^2u^{(2)}(x) + \frac{h^4}{12}u^{(4)}(x)\Big] - \Big[2u(x) + 4h^2u^{(2)}(x) + \frac{16h^4}{12}u^{(4)}(x)\Big] - 30u(x) + O(h^6)\\ = 12h^2u^{(2)}(x) + O(h^6)\,.\end{aligned}$$

This result and the definition of "Big O" indicate that

$$u^{(2)}(x) = \frac{-u(x-2h) + 16u(x-h) - 30u(x) + 16u(x+h) - u(x+2h)}{12h^2} + O(h^4)\,.$$

These finite difference formulae are not used for several obvious reasons. First, nodes neighbouring on a boundary cannot be treated easily even when the value of the solution is prescribed on the boundary (Dirichlet boundary conditions). Second, the underlying system of equations gives rise to a penta-diagonal system which is not so easy to solve. Third, the matrix of this penta-diagonal system of equations is not diagonally dominated. Consequently pivoting may be required.

Solution 5.

The combination of derivatives defined by ψ occurs frequently in problems involving non-uniform material or geometrical properties. The second order central difference expression for ψ may be

written down using the identity

$$\frac{d}{dx}\Big(a(x)\,\frac{df}{dx}\Big) = \frac{da}{dx}\,f(x) + a(x)\,\frac{d^2f}{dx^2} = \frac{1}{2}\Big[\,\frac{d^2}{dx^2}\Big(a(x)f(x)\Big) - f(x)\frac{d^2a}{dx^2} + a(x)\frac{d^2f}{dx^2}\,\Big]\,.$$

Consequently,

$$\begin{aligned}
\psi_k &= \frac{1}{2}\Big[\,\frac{a_{k+1}f_{k+1} - 2a_kf_k + a_{k-1}f_{k-1}}{h^2} - f_k\,\frac{a_{k+1} - 2a_k + a_{k-1}}{h^2} + a_k\,\frac{f_{k+1} - 2f_k + f_{k-1}}{h^2}\,\Big] \\
&= \frac{1}{2h^2}\Big[\,a_{k+1}f_{k+1} - 2a_kf_k + a_{k-1}f_{k-1} - f_ka_{k+1} + 2f_ka_k - f_ka_{k-1} + a_kf_{k+1} - 2a_kf_k + a_kf_{k-1}\,\Big] \\
&= \frac{1}{2h^2}\Big[\,(a_{k+1} + a_k)f_{k+1} - (a_{k+1} + 2a_k + a_{k-1})f_k + (a_{k-1} + a_k)f_{k-1}\,\Big]\,.
\end{aligned}$$

It is obvious from its construction that this finite difference expression for ψ_k is second order accurate.

Solution 6.

The Taylor series expansions of $f(x-h)$ and $f(x+\theta h)$ are

$$\begin{aligned}
f(x-h) &= f(x) - hf'(x) + \frac{h^2}{2}f''(x) + O(h^3) \\
f(x+\theta h) &= f(x) + \theta hf'(x) + \frac{\theta^2h^2}{2}f''(x) + O(h^3)\,.
\end{aligned}$$

Subtracting these expressions gives

$$f(x+\theta h) - f(x-h) = h(1+\theta)f'(x) + \frac{h^2(\theta^2-1)}{2}f''(x) + O(h^3)$$

which in turn leads to the result

$$f'(x) = \frac{f(x+\theta h) - f(x-h)}{(\theta+1)h} + O(h)\,.$$

Solution 7.

Unlike the previous example, there is now no need to eliminate $f(x)$ between the equations

$$\begin{aligned}
f(x-h) &= f(x) - hf'(x) + \frac{h^2}{2}f''(x) + O(h^3) \\
f(x+\theta h) &= f(x) + \theta hf'(x) + \frac{\theta^2h^2}{2}f''(x) + O(h^3)
\end{aligned}$$

since its value is known. The idea is to eliminate the term involving $f''(x)$. Thus

$$f(x+\theta h) - \theta^2 f(x-h) = (1-\theta^2)f(x) + (\theta+\theta^2)hf'(x) + O(h^3)\,.$$

Straight forward algebra now gives

$$f'(x) = \frac{f(x+\theta h) - (1-\theta^2)f(x) - \theta^2 f(x-h)}{\theta(1+\theta)h} + O(h^2)\,.$$

Solution 8.

The Taylor series expansion of $f(x-2h)$, $f(x-h)$ and $f(x+\theta h)$ are

$$\begin{aligned} f(x-2h) &= f(x) - 2hf'(x) + 2h^2 f''(x) - \frac{4h^3}{3} f'''(x) + O(h^4) \\ f(x-h) &= f(x) - hf'(x) + \frac{h^2}{2} f''(x) - \frac{h^3}{6} f'''(x) + O(h^4) \\ f(x+\theta h) &= f(x) + \theta h f'(x) + \frac{\theta^2 h^2}{2} f''(x) + \frac{\theta^3 h^3}{6} f'''(x) + O(h^4)\,. \end{aligned}$$

The value of $f'(x)$ is first eliminated between equations 1 and 2 and between equations 2 and 3 to obtain

$$\begin{aligned} f(x-2h) - 2f(x-h) &= -f(x) + h^2 f''(x) - h^3 f'''(x) + O(h^4) \\ 6(\theta f(x-h) + f(x+\theta h)) &= 6(1+\theta) f(x) + 3(\theta+\theta^2) h^2 f''(x) + (\theta^3 - \theta) h^3 f'''(x) + O(h^4)\,. \end{aligned}$$

The final answer is constructed by multiplying the first of the previous equations by $(\theta^3 - \theta)$ and adding to obtain

$$\begin{aligned} &(\theta^3-\theta)(f(x-2h) - 2f(x-h)) + 6(\theta f(x-h) + f(x+\theta h)) = \\ &\quad 6(1+\theta) f(x) - (\theta^3-\theta) f(x) + 3(\theta+\theta^2) h^2 f''(x) + (\theta^3-\theta) h^2 f''(x) + O(h^4) \end{aligned}$$

and this in turn simplifies to

$$\begin{aligned} &\theta(\theta^2-1) f(x-2h) - 2\theta(\theta^2-4) f(x-h) + 6f(x+\theta h) = \\ &\quad (\theta+1)(6+\theta-\theta^2) f(x) + (2\theta + 3\theta^2 + \theta^3) h^2 f''(x) + O(h^4)\,. \end{aligned}$$

Further simplification gives

$$\begin{aligned} &\theta(\theta^2-1) f(x-2h) - 2\theta(\theta^2-4) f(x-h) + 6f(x+\theta h) = \\ &\quad (\theta+1)(2+\theta)(3-\theta) f(x) + \theta(\theta+1)(\theta+2) h^2 f''(x) + O(h^4)\,. \end{aligned}$$

from which it follows by straight forward algebra that

$$f''(x) = \frac{6f(x+\theta h) + (\theta-3)(\theta+1)(\theta+2) f(x) - 2\theta(\theta^2-4) f(x-h) + \theta(\theta^2-1) f(x-2h)}{\theta(\theta+1)(\theta+2) h^2} + O(h^2)\,.$$

Solution 9.

(a) There are three products of two basis functions that contribute to integrals over the element occupying $[x_j, x_{j+1}]$, namely, $u_j^2(x)$, $u_j(x)u_{j+1}(x)$ and $u_{j+1}^2(x)$. Under the change of variable $x = x_j + (x_{j+1} - x_j)\,\lambda$, it is clear that $u_j(x) = (1-\lambda)$ and $u_{j+1}(x) = \lambda$. Consequently,

$$\begin{aligned}
\int_{x_j}^{x_{j+1}} u_j^2(x)\,dx &= (x_{j+1}-x_j)\int_0^1 (1-\lambda)^2\,d\lambda = \frac{x_{j+1}-x_j}{3}\,,\\
\int_{x_j}^{x_{j+1}} u_j(x)u_{j+1}(x)\,dx &= (x_{j+1}-x_j)\int_0^1 (1-\lambda)\lambda\,d\lambda = \frac{x_{j+1}-x_j}{6}\,,\\
\int_{x_j}^{x_{j+1}} u_{j+1}^2(x)\,dx &= (x_{j+1}-x_j)\int_0^1 \lambda^2\,d\lambda = \frac{x_{j+1}-x_j}{3}\,.
\end{aligned}$$

Since basis functions interior to a segment span two elements and those at the ends of a segment span only one element (see, Figure 1.1), it follows immediately that

$$\int_0^L u_k(x)u_j(x)\,dx = \int_{x_{j-1}}^{x_j} u_k(x)u_j(x)\,dx + \int_{x_j}^{x_{j+1}} u_k(x)u_j(x)\,dx$$

where each integral on the right hand side is present only if the underlying element exists.

(b) There are three possible products of two differentiated basis functions that contribute to integrals over the element occupying $[x_j, x_{j+1}]$, namely,

$$\int_{x_j}^{x_{j+1}} \Big(\frac{du_j}{dx}\Big)^2 dx\,, \qquad \int_{x_j}^{x_{j+1}} \frac{du_j}{dx}\,\frac{du_{j+1}}{dx}\,dx\,, \qquad \int_{x_j}^{x_{j+1}} \Big(\frac{du_{j+1}}{dx}\Big)^2 dx\,.$$

The computation of these integrals is immediate once it is recognised that

$$\frac{du_j}{dx} = \frac{-1}{x_{j+1}-x_j}\,, \qquad \frac{du_{j+1}}{dx} = \frac{1}{x_{j+1}-x_j}\,,$$

The result of these computations is

$$\int_{x_j}^{x_{j+1}} \Big(\frac{du_j}{dx}\Big)^2 dx = \int_{x_j}^{x_{j+1}} \Big(\frac{du_{j+1}}{dx}\Big)^2 dx = \frac{1}{x_{j+1}-x_j}\,, \quad \int_{x_j}^{x_{j+1}} \frac{du_j}{dx}\,\frac{du_{j+1}}{dx}\,dx = -\frac{1}{x_{j+1}-x_j}\,.$$

In the computation of

$$\int_0^L \frac{du_k(x)}{dx}\,\frac{du_j(x)}{dx}\,dx = \int_{x_{j-1}}^{x_j} u_i\,\frac{du_k}{dx}\,\frac{du_j}{dx}\,dx + \int_{x_j}^{x_{j+1}} u_i\,\frac{du_k}{dx}\,\frac{du_j}{dx}\,dx$$

it is again understood that each integral on the right hand side is present only if the underlying element exists.

(c) There are four possible products of three basis functions that contribute to integrals over the element occupying $[x_j, x_{j+1}]$, namely, $u_j^3(x)$, $u_j^2(x)u_{j+1}(x)$, $u_j(x)u_{j+1}^2(x)$ and $u_{j+1}^3(x)$. The

change of variable $x = x_j + (x_{j+1} - x_j)\,\lambda$ now yields.

$$\begin{aligned}
\int_{x_j}^{x_{j+1}} u_j^3(x)\,dx &= (x_{j+1}-x_j)\int_0^1 (1-\lambda)^3\,d\lambda = \frac{x_{j+1}-x_j}{4}\,,\\
\int_{x_j}^{x_{j+1}} u_j^2(x)u_{j+1}(x)\,dx &= (x_{j+1}-x_j)\int_0^1 (1-\lambda)^2\lambda\,d\lambda = \frac{x_{j+1}-x_j}{12}\,,\\
\int_{x_j}^{x_{j+1}} u_j(x)u_{j+1}^2(x)\,dx &= (x_{j+1}-x_j)\int_0^1 (1-\lambda)\lambda^2\,d\lambda = \frac{x_{j+1}-x_j}{12}\,,\\
\int_{x_j}^{x_{j+1}} u_{j+1}^3(x)\,dx &= (x_{j+1}-x_j)\int_0^1 \lambda^3\,d\lambda = \frac{x_{j+1}-x_j}{4}\,.
\end{aligned}$$

Since basis functions interior to a segment span two elements and those at the ends of a segment span only one element, it again follows immediately that

$$\int_0^L u_i(x)u_k(x)u_j(x)\,dx = \int_{x_{j-1}}^{x_j} u_i(x)u_k(x)u_j(x)\,dx + \int_{x_j}^{x_{j+1}} u_i(x)u_k(x)u_j(x)\,dx$$

where each integral on the right hand side is present only if the underlying element exists.

(d) There are four possible products of one basis function and two differentiated basis functions that contribute to integrals over the element occupying $[x_j, x_{j+1}]$, namely,

$$\int_{x_j}^{x_{j+1}} u_j\left(\frac{du_j}{dx}\right)^2 dx\,,\quad \int_{x_j}^{x_{j+1}} u_j\,\frac{du_j}{dx}\,\frac{du_{j+1}}{dx}\,dx\,,\quad \int_{x_j}^{x_{j+1}} u_j\left(\frac{du_{j+1}}{dx}\right)^2 dx\,,$$

$$\int_{x_j}^{x_{j+1}} u_{j+1}\left(\frac{du_j}{dx}\right)^2 dx\,,\quad \int_{x_j}^{x_{j+1}} u_{j+1}\,\frac{du_j}{dx}\,\frac{du_{j+1}}{dx}\,dx\,,\quad \int_{x_j}^{x_{j+1}} u_{j+1}\left(\frac{du_{j+1}}{dx}\right)^2 dx\,.$$

The computation of these integrals is immediate once it is recognised that

$$\begin{aligned}
\frac{du_j}{dx} &= \frac{-1}{x_{j+1}-x_j}\,, & \int_{x_j}^{x_{j+1}} u_j(x)\,dx &= \frac{x_{j+1}-x_j}{2}\,,\\
\frac{du_{j+1}}{dx} &= \frac{1}{x_{j+1}-x_j}\,, & \int_{x_j}^{x_{j+1}} u_{j+1}(x)\,dx &= \frac{x_{j+1}-x_j}{2}\,.
\end{aligned}$$

The result of these computations is

$$\left.\begin{aligned}
\int_{x_j}^{x_{j+1}} u_j\left(\frac{du_j}{dx}\right)^2 dx &= \int_{x_j}^{x_{j+1}} u_{j+1}\left(\frac{du_j}{dx}\right)^2 dx\\
\int_{x_j}^{x_{j+1}} u_j\left(\frac{du_{j+1}}{dx}\right)^2 dx &= \int_{x_j}^{x_{j+1}} u_{j+1}\left(\frac{du_{j+1}}{dx}\right)^2 dx
\end{aligned}\right] = \frac{1}{2}\,\frac{1}{x_{j+1}-x_j}\,,$$

$$\int_{x_j}^{x_{j+1}} u_j\,\frac{du_j}{dx}\,\frac{du_{j+1}}{dx}\,dx = \int_{x_j}^{x_{j+1}} u_{j+1}\,\frac{du_j}{dx}\,\frac{du_{j+1}}{dx}\,dx = -\frac{1}{2}\,\frac{1}{x_{j+1}-x_j}\,.$$

In the computation of

$$\int_0^L u_i(x)\,\frac{du_k(x)}{dx}\,\frac{du_j(x)}{dx}\,dx = \int_{x_{j-1}}^{x_j} u_i\,\frac{du_k}{dx}\,\frac{du_j}{dx}\,dx + \int_{x_j}^{x_{j+1}} u_i\,\frac{du_k}{dx}\,\frac{du_j}{dx}\,dx$$

it is again understood that each integral on the right hand side is present only if the underlying element exists.

Solution 10.

Let $x_j = jL/n$ and define $f_j = \widehat{f}(x_j)$ where $\widehat{f}(x)$ is given by the partial sum

$$\widehat{f}(x) = \sum_{k=-n/2}^{n/2-1} c_k\, e^{2\pi ikx/L}\,, \qquad x \in [0, L]\,.$$

The value f_j is obtained from this summation by replacing x by x_j to obtain

$$f_j = \widehat{f}(x_j) = \sum_{k=-n/2}^{n/2-1} c_k \exp\Big[\frac{2\pi ikjL}{nL}\Big] = \sum_{k=-n/2}^{n/2-1} c_k\, e^{2\pi ikj/n}\,.$$

Since the solution is given *a priori*, there is no need to solve these equations – it is enough to show that the above expression for f_j satisfies

$$c_k = \frac{1}{n}\sum_{j=0}^{n-1} f_j\, e^{-2\pi ijk/n}\,.$$

With this action in mind, consider

$$\begin{aligned}\sum_{j=0}^{n-1} f_j\, e^{-2\pi ijk/n} &= \sum_{j=0}^{n-1}\Big[\sum_{m=-n/2}^{n/2-1} c_m\, e^{2\pi imj/n}\Big] e^{-2\pi ijk/n}\,,\\ &= \sum_{m=-n/2}^{n/2-1} c_m \sum_{j=0}^{n-1} e^{2\pi imj/n}\, e^{-2\pi ijk/n} = \sum_{m=-n/2}^{n/2-1} c_m \sum_{j=0}^{n-1} e^{2\pi i(m-k)j/n}\,.\end{aligned}$$

Now consider the value of the second summation on the right hand side of this equation. If $k = m$ then each term in this summation is one and the value of the sum is n. The same comment applies if $k - m$ is a multiple of n, but this is impossible in this particular problem. So if $k - m \neq 0$ then

$$\sum_{j=0}^{n-1} e^{2\pi i(m-k)j/n} = \frac{1 - e^{2\pi i(m-k)}}{1 - e^{2\pi i(m-k)/n}} = 0$$

since $e^{2\pi i(m-k)} = 1$. In conclusion,

$$\sum_{j=0}^{n-1} e^{2\pi i(m-k)j/n} = \begin{cases} n & k = m \\ 0 & k \neq m \end{cases}$$

and therefore

$$\sum_{j=0}^{n-1} f_j\, e^{-2\pi ijk/n} = \sum_{m=-n/2}^{n/2-1} c_m \sum_{j=0}^{n-1} e^{2\pi i(m-k)j/n} = c_k\,.$$

Solution 11.

The aim is to find $y_1 = y(1/4)$, $y_2 = y(1/2)$ and $y_3 = y(3/4)$ bearing in mind that $y_0 = 0$ and $y_4 = 1$ from the boundary conditions. The central difference formulae indicate that y_1, y_2 and y_3 are the

solutions of the linear equations

$$\begin{bmatrix} -\frac{127}{64} & 1 & 0 \\ 1 & -\frac{63}{32} & 1 \\ 0 & 1 & -\frac{125}{64} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} \frac{1}{16} \\ \frac{1}{16} \\ -\frac{15}{16} \end{bmatrix}.$$

These equations have approximate solution $y_1 = 0.173$, $y_2 = 0.405$ and $y_3 = 0.687$. The Maple code to solve this system of equations is

```
> with(linalg):
> A:=array([[-127/64,1,0],[1,-63/32,1],[0,1,-125/64]]);
> b:=array([1/16,1/16,-15/16]);
> x:=linsolve(A,b);
```

The Maple solution to this problem is

$$x = \left[\frac{83572}{484029}, \frac{196090}{484029}, \frac{332732}{484029} \right].$$

Solution 12.

The aim is to find $y_1 = y(1/4)$, $y_2 = y(1/2)$, $y_3 = y(3/4)$ and $y_4 = y(1)$ bearing in mind that only $y_0 = 0$ is known from the boundary conditions. The central difference formulae indicate that y_1, y_2, y_3 and y_4 are the solutions of the linear equations

$$\begin{bmatrix} -\frac{31}{16} & 1 & 0 & 0 \\ 1 & -\frac{31}{16} & 1 & 0 \\ 0 & 1 & -\frac{31}{16} & 1 \\ 0 & 1 & -4 & 3 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

These equations have approximate solution $y_1 = 1.354$, $y_2 = 1.623$, $y_3 = 1.791$ and $y_4 = 1.847$. The Maple code to solve this system of equations is

```
> with(linalg):
> A:=array([[-31/16,1,0,0],[1,-31/16,1,0],[0,1,-31/16,1],[0,1,-4,3]]);
> b:=array([-1,0,0,0]);
> x:=linsolve(A,b);
```

The Maple solution to this problem is

$$\begin{aligned} x &= \left[\frac{6192}{4573}, \frac{7424}{4573}, \frac{8192}{4573}, \frac{8448}{4573}\right] \\ &\approx [1.354034551,\ 1.623441942,\ 1.791384212,\ 1.847364968]\,. \end{aligned}$$

It is straight forward to demonstrate that

$$y(x) = \frac{\cos(1-x)}{\cos 1}$$

satisfies the differential equation and the boundary conditions. The exact values of y_1, y_2, y_3 and y_4 are compared with the approximate solution to get

	y_1	y_2	y_3	y_4
Exact	1.354221259	1.624243599	1.793278339	1.850815718
Approx	1.354034551	1.623441942	1.791384212	1.847364968

Hand calculation The hand calculation of the solution to this problem working to an accuracy of 3dp involves the manipulation of the augmented matrix as follows:

$$\begin{bmatrix} -1.936 & 1.000 & 0.000 & 0.000 & -1.000 \\ 1.000 & -1.938 & 1.000 & 0.000 & 0.000 \\ 0.000 & 1.000 & -1.938 & 1.000 & 0.000 \\ 0.000 & 1.000 & -4.000 & 3.000 & 0.000 \end{bmatrix}$$

$$\sim \begin{bmatrix} -1.936 & 1.000 & 0.000 & 0.000 & -1.000 \\ 1.000 & -1.938 & 1.000 & 0.000 & 0.000 \\ 0.000 & 1.000 & -1.938 & 1.000 & 0.000 \\ 0.000 & 0.000 & -2.062 & 2.000 & 0.000 \end{bmatrix}$$

$$\sim \begin{bmatrix} -1.936 & 1.000 & 0.000 & 0.000 & -1.000 \\ 0.000 & -1.422 & 1.000 & 0.000 & -0.516 \\ 0.000 & 1.000 & -1.938 & 1.000 & 0.000 \\ 0.000 & 0.000 & -2.062 & 2.000 & 0.000 \end{bmatrix}$$

$$\sim \begin{bmatrix} -1.936 & 1.000 & 0.000 & 0.000 & -1.000 \\ 0.000 & -1.422 & 1.000 & 0.000 & -0.516 \\ 0.000 & 0.000 & -1.235 & 1.000 & -0.363 \\ 0.000 & 0.000 & -2.062 & 2.000 & 0.000 \end{bmatrix}$$

$$\sim \begin{bmatrix} -1.936 & 1.000 & 0.000 & 0.000 & -1.000 \\ 0.000 & -1.422 & 1.000 & 0.000 & -0.516 \\ 0.000 & 0.000 & -2.062 & 2.000 & 0.000 \\ 0.000 & 0.000 & -1.235 & 1.000 & -0.363 \end{bmatrix}$$

The final step of this calculation gives the upper triangular form

$$\begin{bmatrix} -1.936 & 1.000 & 0.000 & 0.000 & -1.000 \\ 0.000 & -1.422 & 1.000 & 0.000 & -0.516 \\ 0.000 & 0.000 & -2.062 & 2.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & -0.198 & -0.363 \end{bmatrix}$$

which in turn leads to the solutions

$$y_4 = 1.833\,, \quad y_3 = 1.778\,, \quad y_2 = 1.613\,, \quad y_1 = 1.348\,.$$

Solution 13.

The dissection has nodes $x_0 = 1$, $x_1 = 4/3$, $x_2 = 5/3$, and $x_3 = 2$. The boundary conditions indicate that $y_0 = 0$ and $y_3 = 4$. The remaining unknowns y_1, y_2 are the subject of the linear equations

$$\begin{bmatrix} -31 & 14 \\ \frac{55}{2} & -49 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} \frac{16}{9} \\ -\frac{785}{9} \end{bmatrix}$$

with augmented matrix

$$B = \begin{bmatrix} -31 & 14 & \frac{16}{9} \\ \frac{55}{2} & -49 & -\frac{785}{9} \end{bmatrix} \sim \begin{bmatrix} -31 & 14 & \frac{16}{9} \\ 0 & -\frac{1134}{31} & -\frac{2655}{31} \end{bmatrix}.$$

Using back substitution, the solutions are

$$y_2 = \frac{2655}{1134} = \frac{295}{126} \approx 2.3412\,, \quad y_1 \approx 1.0000\,.$$

The exact solution to four decimal places is $y(4/3) = 0.9978$ and $y(5/3) = 2.3394$.

Solution 14.

Forward difference The dissection has nodes $x_0 = 1$, $x_1 = 4/3$, $x_2 = 5/3$, and $x_3 = 2$. The boundary conditions indicate that only $y_4 = 1$ is known. The remaining unknowns y_0, y_1 and y_2 are the subject of the linear equations

$$\begin{bmatrix} -3 & 4 & -1 \\ 18 & -31 & 14 \\ 0 & \frac{55}{2} & -49 \end{bmatrix} \begin{bmatrix} y_0 \\ y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 0 \\ \frac{16}{9} \\ -\frac{355}{18} \end{bmatrix}$$

in which the forward difference formula has been used for $y'(1)$. These equations have approximate solution $y_0 = 0.524$ $y_1 = 0.575$ and $y_2 = 0.725$. The Maple code to solve this system of equations is

```
> with(linalg):
> A:=array([[-3,4,-1],[18,-31,14],[0,55/2,-49]]);
> b:=array([0,16/9,-355/18]);
> x:=linsolve(A,b);
```

The Maple solution to this problem is

$$x = \left[\frac{43}{82}, \frac{212}{369}, \frac{535}{738} \right].$$

The exact values of y_0, y_1 and y_2 are compared with the approximate solution to get

	y_0	y_1	y_2
Exact	0.629445676	0.658688936	0.772916230
Approx	0.524390244	0.574525745	0.724932249

Fictitious nodes The second and third equations connecting y_0, y_1 and y_2 are identical to those derived in the forward difference approach. The fictitious nodes procedure creates the fictitious solution y_{-1} to be determined from the equations

$$y_{-1} = y_1\,, \quad \frac{y_{-1} - 2y_0 + y_1}{h^2} + y_0 = 1\,, \qquad \rightarrow \qquad -17y_0 + 18y_1 = 1\,.$$

The unknowns y_0, y_1 and y_2 therefore satisfy the linear equations

$$\begin{bmatrix} -17 & 18 & 0 \\ 18 & -31 & 14 \\ 0 & \frac{55}{2} & -49 \end{bmatrix} \begin{bmatrix} y_0 \\ y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 \\ \frac{16}{9} \\ -\frac{355}{18} \end{bmatrix}$$

where the fictitious nodes procedure has been used for $y'(1)$. These equations have approximate solution $y_0 = 0.667$, $y_1 = 0.685$ and $y_2 = 0.787$. The Maple code to solve this system of equations is

```
> with(linalg):
> A:=array([[-17,18,0],[18,-31,14],[0,55/2,-49]]);
> b:=array([1,16/9,-355/18]);
> x:=linsolve(A,b);
```

The Maple solution to this problem is

$$x = \left[\frac{2}{3}, \frac{37}{54}, \frac{85}{108} \right].$$

The exact values of y_0, y_1 and y_2 are compared with the approximate solution to get

	y_0	y_1	y_2
Exact	0.629445676	0.658688936	0.772916230
Approx	0.666666667	0.685185185	0.787037037

Solution 15.

From the solution to question 1, Taylors theorem asserts that

$$u''(x) = \frac{u(x+h) - 2u(x) + u(x-h)}{h^2} - \frac{h^2}{12}\, u''''(\xi) + O(h^6)\,.$$

If $u_{xx} = f(x)$ then it is clear from this identity that

$$f(x) = \frac{u(x+h) - 2u(x) + u(x-h)}{h^2} - \frac{h^2}{12}\, f''(x) + O(h^4)$$

which in turn gives

$$\begin{aligned} \frac{u(x+h) - 2u(x) + u(x-h)}{h^2} &= f(x) + \frac{h^2}{12}\left[\, \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} + O(h^2)\,\right] + O(h^4) \\ &= \frac{1}{12}\left[f(x-h) + 10f(x) + f(x+h)\right] + O(h^4)\,. \end{aligned}$$

The equations written out in the question may be derived directly from this identity. In particular, it is clear that this technique may be applied to any second order differential equations of type $u_{xx} = f(x, u, u_x)$, and will allow it to be solved to higher accuracy than might initial be imagined.

Solution 16.

Let $\phi(h) = u(x+h, y+h) - u(x+h, y-h) - u(x-h, y+h) + u(x-h, y-h)$ then simple algebra gives

$$\phi(h) = 2h\left[\, \frac{u(x+h, y+h) - u(x+h, y-h)}{2h} - \frac{u(x-h, y+h) - u(x-h, y-h)}{2h}\,\right]$$

Each fraction is now expanded as a power series about y using the idea that

$$\frac{u(z, y+h) - u(z, y-h)}{2h} = u_y(z, y) - \frac{h^2}{6}\, u_{yyy}(z, \eta)$$

The result of this operation is that

$$\phi(h) = 2h\left[\, u_y(x+h, y) - \frac{h^2}{6}\, u_{yyy}(x+h, \eta_1) - u_y(x-h, y) + \frac{h^2}{6}\, u_{yyy}(x-h, \eta_2)\,\right]$$

Applying the same idea to $u_y(x+h, y) - u_y(x-h, y)$ yields

$$u_y(x+h, y) - u_y(x-h, y) = 2h u_{xy}(x, y) - \frac{h^3}{6} u_{xxxy}(\xi_1, y)\,.$$

This formula is now used to simplify ϕ to obtain

$$\phi(h) = 4h^2\, u_{xy}(x,y) + O(h^4)$$

from which it is transparent that

$$\frac{\partial^2 u(x,y)}{\partial x \partial y} = \frac{u(x+h,y+h) - u(x+h,y-h) - u(x-h,y+h) + u(x-h,y-h)}{4h^2} + O(h^2)\,.$$

Solution 17.

Under the change of variable $x = e^t$, it is easy to see that $xu_x = u_t$. The original equation can be re-expressed in the form $x((xu)_x)_x + u = x$ which then becomes $u_{tt} + u = e^t$ under the given change of variable. The points $x = \varepsilon$ and $x = 1$ become respectively $t = \log\varepsilon$ and $t = 0$. The general solution for $u(t)$ is

$$u(t) = A\sin t + B\cos t + \frac{e^t}{2}$$

where A and B are determined by the conditions $u(\log\varepsilon) = u(0) = 0$. The discretised equation for u is

$$\frac{u_{k+1} - 2u_k + u_{k-1}}{h^2} + u_k = e^{t_k} \quad \rightarrow \quad u_{k+1} - (2-h^2)u_k + u_{k-1} = h^2 e^{t_k}\,.$$

where $u_0 = u_n = 0$. The unknowns values $u_1, \cdots, u_{n-1}$ satisfy the linear system

$$\begin{aligned}
-(2-h^2)u_1 + u_2 &= h^2 e^{t_0} \\
u_1 - (2-h^2)u_2 + u_3 &= h^2 e^{t_1} \\
\vdots \qquad\qquad &\vdots \quad \vdots \\
u_{k-1} - (2-h^2)u_k + u_{k+1} &= h^2 e^{t_k} \\
\vdots \qquad\qquad &\vdots \quad \vdots \\
u_{n-2} - (2-h^2)u_{n-1} &= h^2 e^{t_{n-1}}\,.
\end{aligned}$$

This approach is superior because the new differential equation is not a singular equation.

Solution 18.

Clearly $u = \sin x$ satisfies the initial conditions and the differential equation.

In the usual notation, the central difference representation of the differential equation is

$$\frac{u_{k+1} - 2u_k + u_{k-1}}{h^2} + 2u_k\,\frac{u_{k+1} - u_{k-1}}{2h} + u_k = \sin 2x_k$$

where $u_0 = u_n = 0$. This equation further simplifies to

$$u_{k+1} - 2u_k + u_{k-1} + hu_k\,(u_{k+1} - u_{k-1}) + h^2 u_k = h^2 \sin 2x_k\,.$$

Taking account of the boundary conditions, the unknowns $u_1, \cdots, u_{n-1}$ now satisfy $F_1 = F_2 = \cdots = F_{n-1} = 0$ where

$$\begin{aligned} F_1 &= u_2 - 2u_1 + hu_1\, u_2 + h^2 u_1 - h^2 \sin 2x_1 \\ &\vdots \\ F_k &= u_{k+1} - 2u_k + u_{k-1} + hu_k\,(u_{k+1} - u_{k-1}) + h^2 u_k - h^2 \sin 2x_k \\ &\vdots \\ F_{n-1} &= -2u_{n-1} + u_{n-2} - hu_{n-1}\, u_{n-2} + h^2 u_{n-1} - h^2 \sin 2x_{n-1}\,. \end{aligned}$$

The Jacobian matrix of F is

$$J = \begin{bmatrix} hu_2 + h^2 - 2 & 1 + hu_1 & 0 & \cdots & \cdots & \cdots & 0 \\ 1 - hu_2 & \begin{matrix} h(u_3 - u_1) \\ +h^2 - 2 \end{matrix} & 1 + hu_2 & 0 & \cdots & \cdots & 0 \\ 0 & 1 - hu_3 & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & 1 - hu_k & \begin{matrix} h(u_{k+1} - u_{k-1}) \\ +h^2 - 2 \end{matrix} & 1 + hu_k & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & 0 & 1 - hu_{n-1} & \begin{matrix} -hu_{n-2} \\ +h^2 - 2 \end{matrix} \end{bmatrix}$$

Another possible approach is to rewrite the equations $F_1 = F_2 = \cdots = F_{n-1} = 0$ in the iterative form

$$\begin{aligned} u_1 &= \frac{u_2 + hu_1\, u_2 - h^2 \sin 2x_1}{2 - h^2} \\ &\vdots \\ u_k &= \frac{u_{k+1} + u_{k-1} + hu_k\,(u_{k+1} - u_{k-1}) - h^2 \sin 2x_k}{2 - h^2} \\ &\vdots \\ u_{n-1} &= \frac{u_{n-2} - hu_{n-1}\, u_{n-2} - h^2 \sin 2x_{n-1}}{2 - h^2} \end{aligned}$$

Solution 19.

(a) To establish this result first recognise that

$$u(x-h,y) + u(x+h,y) = 2u(x,y) + h^2\,\frac{\partial^2 u(x,y)}{\partial x^2} + \frac{h^4}{12}\,\frac{\partial^4 u(x,y)}{\partial x^4} + O(h^6)\,.$$

This result is now used in with y replaced by $y+h$ and $y-h$ to get

$$\begin{aligned}
&[u(x-h,y+h)+u(x+h,y+h)]+[u(x-h,y-h)+u(x+h,y-h)] \\
&= 2u(x,y+h)+2u(x,y-h)+h^2\,\frac{\partial^2 u(x,y+h)}{\partial x^2}+h^2\,\frac{\partial^2 u(x,y-h)}{\partial x^2} \\
&\quad +\frac{h^4}{12}\,\frac{\partial^4 u(x,y+h)}{\partial x^4}+\frac{h^4}{12}\,\frac{\partial^4 u(x,y-h)}{\partial x^4}+O(h^6)\,.
\end{aligned} \tag{6.1}$$

Taylors theorem is now used to assert that

$$\begin{aligned}
u(x,y-h)+u(x,y+h) &= 2u(x,y)+h^2\,\frac{\partial^2 u(x,y)}{\partial y^2}+\frac{h^4}{12}\,\frac{\partial^4 u(x,y)}{\partial y^4}+O(h^6) \\
\frac{\partial^2 u(x,y+h)}{\partial x^2}+\frac{\partial^2 u(x,y-h)}{\partial x^2} &= 2\frac{\partial^2 u(x,y)}{\partial x^2}+h^2\,\frac{\partial^4 u}{\partial x^2\partial y^2}+O(h^4) \\
\frac{\partial^4 u(x,y+h)}{\partial x^4}+\frac{\partial^4 u(x,y-h)}{\partial x^4} &= 2\frac{\partial^4 u}{\partial x^4}+O(h^2)\,.
\end{aligned}$$

These results are now inserted into (6.1) to obtain

$$\begin{aligned}
&u(x-h,y+h)+u(x+h,y+h)-4u(x,y)+u(x-h,y-h)+u(x+h,y-h) \\
&= 2h^2\left(\frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 u}{\partial x^2}\right)+\frac{h^4}{6}\left(\frac{\partial^4 u}{\partial y^4}+6\frac{\partial^4 u}{\partial x^2\partial y^2}+\frac{\partial^4 u}{\partial x^4}\right)+O(h^6)
\end{aligned} \tag{6.2}$$

where all derivative are evaluated at (x,y). The immediate conclusion from this calculation is

$$\begin{aligned}
\frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 u}{\partial x^2} &= \frac{1}{2h^2}\Big[u(x-h,y+h)+u(x+h,y+h)-4u(x,y) \\
&\qquad +u(x-h,y-h)+u(x+h,y-h)\Big]+O(h^2)\,.
\end{aligned} \tag{6.3}$$

In particular, if u satisfies Laplace's equation then $u_{xxxx}+2u_{xxyy}+u_{yyyy}=0$. This simplification leads to the particular result

$$\begin{aligned}
\frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 u}{\partial x^2} &= \frac{1}{2h^2}\Big[u(x-h,y+h)+u(x+h,y+h)-4u(x,y) \\
&\qquad +u(x-h,y-h)+u(x+h,y-h)\Big]+\frac{2h^2}{3}\,\frac{\partial^4 u}{\partial x^2\partial y^2}+O(h^4)\,.
\end{aligned} \tag{6.4}$$

(b) This result is a combination of result (a) and the standard finite difference approximation for the Laplacian of u. Consider first the Taylor series approximations used in the derivation of the standard central difference formula for the Laplacian of u. These are

$$\begin{aligned}
u(x-h,y)+u(x+h,y) &= 2u(x,y)+h^2\,\frac{\partial^2 u}{\partial x^2}+\frac{h^4}{12}\,\frac{\partial^4 u}{\partial x^4}+O(h^6)\,, \\
u(x,y-h)+u(x,y+h) &= 2u(x,y)+h^2\,\frac{\partial^2 u}{\partial y^2}+\frac{h^4}{12}\,\frac{\partial^4 u}{\partial y^4}+O(h^6)\,.
\end{aligned}$$

These results are added together to give

$$\begin{aligned} &u(x-h,y)+u(x+h,y)-4u(x,y)+u(x,y-h)+u(x,y+h) \\ &\qquad = h^2\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)+\frac{h^4}{12}\left(\frac{\partial^4 u}{\partial x^4}+\frac{\partial^4 u}{\partial y^4}\right)+O(h^6) \end{aligned}$$

where all derivative are evaluated at (x, y). The immediate conclusion from this calculation is

$$\begin{aligned} \frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 u}{\partial x^2} &= \frac{1}{h^2}\Big[u(x-h,y)+u(x,y+h)-4u(x,y) \\ &\qquad +u(x,y-h)+u(x+h,y)\Big]+O(h^2)\,. \end{aligned} \tag{6.5}$$

When u satisfies Laplace's equation then $u_{xxxx}+2u_{xxyy}+u_{yyyy}=0$ and equation (6.5) takes on the particular form

$$\begin{aligned} \frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 u}{\partial x^2} &= \frac{1}{h^2}\Big[u(x-h,y)+u(x,y+h)-4u(x,y) \\ &\qquad +u(x,y-h)+u(x+h,y)\Big]-\frac{h^2}{6}\frac{\partial^4 u}{\partial x^2\partial y^2}+O(h^4)\,. \end{aligned} \tag{6.6}$$

The result is quoted in the question is constructed by multiplying equation (6.3) by $2h^2$ and adding to that equation (6.6) multiplied by $4h^2$. The truncation error is $O(h^2)$ in general and $O(h^4)$ if u satisfies Laplace's equation.

Solution 20.

It is required to find the eigenvalues of $H_{\rm GJ}=-D^{-1}(L+U)$ and $H_{\rm GS}=-(D+L)^{-1}U$ when

$$A=L+D+U=\begin{bmatrix} 1 & 2 & 4 \\ 1/8 & 1 & 1 \\ -1 & 4 & 1 \end{bmatrix}.$$

Gauss-Jacobi

If $H_{\rm GJ}X=\lambda X$ then $(D\lambda+L+U)X=0$. Thus λ satisfies the equation

$$\begin{vmatrix} \lambda & 2 & 4 \\ 1/8 & \lambda & 1 \\ -1 & 4 & \lambda \end{vmatrix} = \lambda^3-\frac{\lambda}{4}=0\,.$$

The eigenvalues are $\lambda=0$ and $\lambda=\pm 1/2$. All eigenvalues lie within the unit circle and so the Gauss-Jacobi algorithm converges for these equations.

Gauss-Seidel

If $H_{\rm GS}X = \lambda X$ then $(D\lambda + \lambda L + U)X = 0$. Thus λ satisfies the equation

$$\begin{vmatrix} \lambda & 2 & 4 \\ \lambda/8 & \lambda & 1 \\ -\lambda & 4\lambda & \lambda \end{vmatrix} = \lambda^3 + \frac{7\lambda^2}{4} - 2\lambda = 0\,.$$

The eigenvalues are $\lambda = 0$ and $\lambda = (-7 \pm \sqrt{177})/8$. Not all eigenvalues lie within the unit circle and so the Gauss-Seidel algorithm diverges for these equations.

Solution 21.

This is a repetition of the previous problem. It is required to find the eigenvalues of $H_{\rm GJ} = -D^{-1}(L+U)$ and $H_{\rm GS} = -(D+L)^{-1}U$ when

$$A = L + D + U = \begin{bmatrix} 3 & 2 & 1 \\ 2 & 3 & 2 \\ 1 & 2 & 3 \end{bmatrix}\,.$$

Gauss-Jacobi

If $H_{\rm GJ}X = \lambda X$ then $(D\lambda + L + U)X = 0$. Thus λ satisfies the equation

$$\begin{vmatrix} 3\lambda & 2 & 1 \\ 2 & 3\lambda & 2 \\ 1 & 2 & 3\lambda \end{vmatrix} = 27\lambda^3 - 27\lambda + 8 = 0\,.$$

This cubic factorises into the product $(3\lambda - 1)(9\lambda^2 + 3\lambda - 8)$. Consequently, the eigenvalues of $H_{\rm GJ}$ are $\lambda = 1/3$ and $\lambda = (-1 \pm \sqrt{33})/6$. The spectral radius is $\rho = (1 + \sqrt{33})/6 > 1$ and therefore not all eigenvalues lie within the unit circle. The Gauss-Jacobi algorithm diverges for these equations.

Gauss-Seidel

If $H_{\rm GS}X = \lambda X$ then $(D\lambda + \lambda L + U)X = 0$. Thus λ satisfies the equation

$$\begin{vmatrix} 3\lambda & 2 & 1 \\ 2\lambda & 3\lambda & 2 \\ \lambda & 2\lambda & 3\lambda \end{vmatrix} = 27\lambda^3 - 23\lambda^2 + 4\lambda = 0\,.$$

The eigenvalues are $\lambda = 0$ and $\lambda = (23 \pm \sqrt{97})/54$. The spectral radius is $\rho = (23 + \sqrt{97})/54 < 1$ and therefore all eigenvalues lie within the unit circle. The Gauss-Seidel algorithm converges for these equations.

Solution 22.

If $H_{\rm GS}X = \lambda X$ then $(D\lambda + \lambda L + U)X = 0$. Thus λ satisfies the equation

$$\begin{vmatrix} \lambda & c & 1 \\ c\lambda & \lambda & c \\ -c^2\lambda & c\lambda & \lambda \end{vmatrix} = \lambda^3 - c^4\lambda = 0\,.$$

The eigenvalues are $\lambda = 0$ and $\lambda = \pm c^2$. The spectral radius is $\rho = c^2$ and therefore the Gauss-Seidel algorithm converges for these equations provided $|\,c\,| < 1$.

The Gauss-Jacobi algorithm for these equations with $c = 2$ requires that the solutions of the equation

$$\begin{vmatrix} \lambda & 2 & 1 \\ 2 & \lambda & 2 \\ -4 & 2 & \lambda \end{vmatrix} = \lambda^3 - 4\lambda - 12 = 0$$

lie within the unit circle. If $g(\lambda) = \lambda^3 - 4\lambda - 12$ then $g(1) = -15$ and $g(\lambda) \to \infty$ as $\lambda \to \infty$. Clearly there is a zero outside the unit circle and so the Gauss-Jacobi algorithm diverges for these equations.

Solution 23.

Consider the matrix identity

$$\begin{bmatrix} 1 & 0 \\ \dfrac{V}{\alpha} & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & C_1 - \dfrac{VW^{\mathrm{T}}}{\alpha} \end{bmatrix} \begin{bmatrix} \alpha & W^{\mathrm{T}} \\ 0 & I \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \dfrac{V}{\alpha} & C_1 - \dfrac{VW^{\mathrm{T}}}{\alpha} \end{bmatrix} \begin{bmatrix} \alpha & W^{\mathrm{T}} \\ 0 & I \end{bmatrix} = \begin{bmatrix} \alpha & W^{\mathrm{T}} \\ V & C_1 \end{bmatrix}.$$

Suppose that $C_1 - VW^{\mathrm{T}}/\alpha = L_1U_1$, then it follows directly from the identity that

$$\begin{bmatrix} \alpha & W^{\mathrm{T}} \\ V & C_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \dfrac{V}{\alpha} & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & L_1U_1 \end{bmatrix} \begin{bmatrix} \alpha & W^{\mathrm{T}} \\ 0 & I \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \dfrac{V}{\alpha} & L_1 \end{bmatrix} \begin{bmatrix} \alpha & W^{\mathrm{T}} \\ 0 & U_1 \end{bmatrix}.$$

In particular, if A^{T} is diagonally dominated then the first row of A is the pivotal row. The task is now to show that if A^{T} is diagonally dominated then so is C_2^{T} where $C_2 = L_1U_1$. Now

$$\begin{aligned}
\sum_{i=1\ i\neq j}^{n-1} |\,(C_2)_{ij}\,| &= \sum_{i=1\ i\neq j}^{n-1} |\,(C_1)_{ij} - v_iw_j/\alpha\,| \\
&\leq \sum_{i=1\ i\neq j}^{n-1} |\,(C_1)_{ij}\,| + \Big|\frac{w_j}{\alpha}\Big| \sum_{i=1\ i\neq j}^{n-1} |\,v_i\,| \\
&= \Big(\sum_{i=1\ i\neq j+1}^{n} |\,A_{i,j+1}\,| - |\,w_j\,|\Big) + \Big|\frac{w_j}{\alpha}\Big|\Big(\sum_{i=1}^{n-1} |\,v_i\,| - |\,v_j\,|\Big) \\
&\leq \Big(|\,(C_1)_{jj}\,| - |\,w_j\,|\Big) + \Big|\frac{w_j}{\alpha}\Big|\Big(|\,\alpha\,| - |\,v_j\,|\Big) \\
&= |\,(C_1)_{jj}\,| - \Big|\frac{w_jv_j}{\alpha}\Big| \\
&\leq \Big|(C_1)_{jj} - \frac{w_jv_j}{\alpha}\Big| = |\,(C_2)_{jj}\,|
\end{aligned}$$

where this last inequality is an application of the identity $|a| - |b| \leq ||a| - |b|| \leq |a - b|$. Thus the matrix C_2 is diagonally dominated. This analysis forms the basis of an induction assumption to demonstrate that k^{th} iteration of the LU decomposition of A may take the k^{th} row as pivotal row - in effect, no pivoting is required in the LU factorisation of A provided A^{T} is diagonally dominated.

Solution 24.

(a) To appreciate why each diagonal entry of a symmetric positive definite matrix is positive, it is enough to note that every positive definite matrix can be represented in the form LDL^{T} where the entries of D are the eigenvalues of A and L is a lower triangular matrix. [Of course, D and L in the representation LDL^{T} are different from their usage in the main body of this solution.] Furthermore, $LDL^{\mathrm{T}} = (LD^{1/2})(LD^{1/2})^{\mathrm{T}} = L_1 L_1^{\mathrm{T}}$. Thus

$$A_{ii} = \sum_{k=1}^{n} (L_1)_{ik}(L_1^{\mathrm{T}})_{ki} = \sum_{k=1}^{n} (L_1)_{ik}(L_1)_{ik} > 0\,.$$

Thus each diagonal entry of A is positive.

(b) Since $A = L + D + U$ is symmetric then $U = L^{\mathrm{T}}$. Take $D = S^2$ and construct $H_1 = SHS^{-1}$. Clearly H and H_1 are similar matrices and therefore have the same eigenvalues. Also

$$\begin{aligned} H_1 = SHS^{-1} = S(D+L)^{-1}L^{\mathrm{T}}S^{-1} &= S(S^2+L)^{-1}L^{\mathrm{T}}S^{-1} \\ &= S[S(I+S^{-1}LS^{-1})S]^{-1}L^{\mathrm{T}}S^{-1} \\ &= SS^{-1}(I+S^{-1}LS^{-1})^{-1}S^{-1}L^{\mathrm{T}}S^{-1} \\ &= (I+S^{-1}LS^{-1})^{-1}S^{-1}L^{\mathrm{T}}S^{-1} \\ &= (I+L_1)^{-1}L_1^{\mathrm{T}} \end{aligned}$$

where $L_1 = S^{-1}LS^{-1}$.

(c) Furthermore, $D^{-1/2}AD^{-1/2} = S^{-1}(D+L+L^{\mathrm{T}})S^{-1}$. By writing $D = S^2$, it follows that

$$D^{-1/2}AD^{-1/2} = I + S^{-1}LS^{-1} + S^{-1}L^{\mathrm{T}}S^{-1} = I + L_1 + L_1^{\mathrm{T}}\,.$$

(d) To show that the Gauss-Seidel algorithm always converges, it is required to show that all the eigenvalues of $H = (D+L)^{-1}U = (D+L)^{-1}L^{\mathrm{T}}$ lie within the unit circle. Let X be an eigenvector of H_1 with eigenvalue λ then $H_1X = \lambda X$. Therefore,

$$(I+L_1)^{-1}L_1^{\mathrm{T}}X = \lambda X \quad \rightarrow \quad L_1^{\mathrm{T}}X = \lambda(I+L_1)X \quad \rightarrow \quad X^{\mathrm{H}}L_1^{\mathrm{T}}X = \lambda X^{\mathrm{H}}(I+L_1)X$$

where X^{H} implies transposition of the complex conjugate of X. Take $Y = X/|\,X\,|$ then it follows that $Y^{\mathrm{H}}L_1^{\mathrm{T}}Y = \lambda + \lambda Y^{\mathrm{H}}L_1 Y$. Let $Y^{\mathrm{H}}L_1^{\mathrm{T}}Y = a + ib$ then

$$\lambda = \frac{a-ib}{1+a+ib} \quad \rightarrow \quad |\,\lambda\,|^2 - 1 = -\frac{1+2a}{1+2a+a^2+b^2}\,.$$

Returning to result (c), it follows immediately that

$$X^{\mathrm{H}}D^{-1/2}AD^{-1/2}X = X^{\mathrm{H}}(I+L_1+L_1^{\mathrm{T}})X = X^{\mathrm{H}}X + X^{\mathrm{H}}L_1X + X^{\mathrm{H}}L_1^{\mathrm{T}}X > 0\,.$$

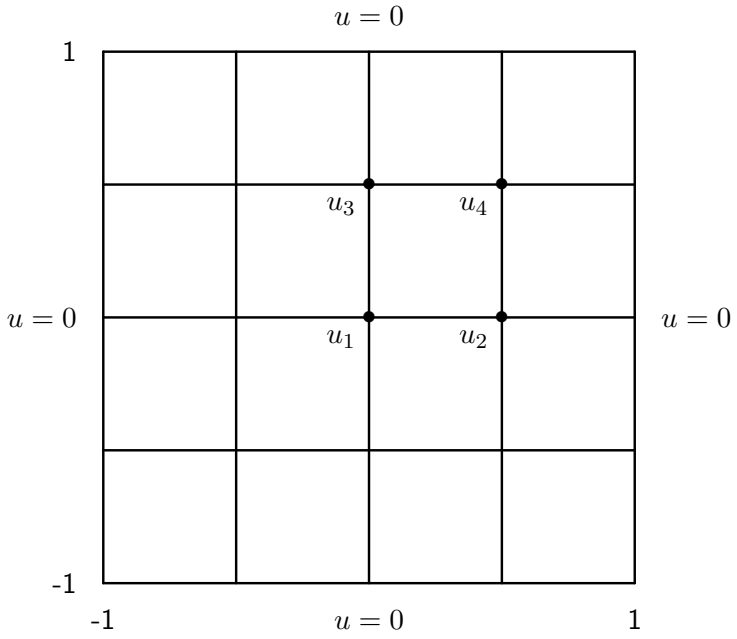
On dividing by $X^{\mathrm{H}}X$, we obtain

$$1 + Y^{\mathrm{H}}L_1Y + Y^{\mathrm{H}}L_1^{\mathrm{T}}Y = 1 + (a + ib) + (a - ib) = 1 + 2a > 0$$

and therefore $|\lambda|^2 - 1 < 0$, leading to $|\lambda| < 1$. Thus all the eigenvalues of H_1 and so H lie within the unit circle which in turn ensures that the Gauss-Seidel algorithm always converges if A is a symmetric and positive definite matrix.

Solution 25.

The region in which the solution is sought consists of sixteen squares as illustrated in the figure.



The discretised form of the partial differential equation is

$$u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j} = -2h^2$$

where $h = 1/2$ where $-2 \leq i \leq 2$ and $-2 \leq j \leq 2$. We consider each internal node in turn.

Node	Position	Equation
1	$(0, 0)$	$u_{1,0} + u_{-1,0} + u_{0,1} + u_{0,-1} - 4u_{0,0} = -2h^2$
2	$(1/2, 0)$	$u_{2,0} + u_{0,0} + u_{1,1} + u_{1,-1} - 4u_{1,0} = -2h^2$
4	$(0, 1/2)$	$u_{1,1} + u_{-1,1} + u_{0,2} + u_{0,0} - 4u_{0,1} = -2h^2$
5	$(1/2, 1/2)$	$u_{2,1} + u_{0,1} + u_{1,2} + u_{1,0} - 4u_{1,1} = -2h^2$

Furthermore, symmetry requires that $u_1 = u_{0,0}$, $u_2 = u_{1,0} = u_{-1,0}$, $u_3 = u_{0,1} = u_{0,-1}$ and $u_4 = u_{1,1} = u_{-1,1} = u_{1,-1}$. The boundary conditions are now introduced into the finite difference equations to

obtain

Node	Equation
1	$2u_2 + 2u_3 - 4u_1 = -2h^2$
2	$u_1 + 2u_4 - 4u_2 = -2h^2$
4	$2u_4 + u_1 - 4u_3 = -2h^2$
5	$u_3 + u_2 - 4u_4 = -2h^2$

It is clear from the second and third equations that $u_2 = u_3$. With this simplification, u_1, u_2 and u_4 satisfy

$$u_1 - u_2 = 1/8\,, \qquad u_1 - 4u_2 + 2u_4 = -1/2\,, \qquad u_2 - 2u_4 = -1/4\,.$$

Eliminating u_1 between equations 1 and 2 gives

$$\left.\begin{aligned} -3u_2 + 2u_4 &= -5/8 \\ u_2 - 2u_4 &= -1/4 \end{aligned}\right] \quad \rightarrow \quad \begin{aligned} u_1 &= \frac{9}{16} \\ u_2 &= \frac{7}{16} \\ u_4 &= \frac{11}{32}\,. \end{aligned}$$

Solution 26.

The discretised form of the partial differential equation is

$$u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j} = h^2 u_{i,j}^3$$

where $h = 1/2$. We consider each internal node in turn.

Node	Position	Equation
1	$(0, 0)$	$u_{1,0} + u_{-1,0} + u_{0,1} + u_{0,-1} - 4u_{0,0} = h^2 u_{0,0}^3$
2	$(1/2, 0)$	$u_{2,0} + u_{0,0} + u_{1,1} + u_{1,-1} - 4u_{1,0} = h^2 u_{1,0}^3$
4	$(0, 1/2)$	$u_{1,1} + u_{-1,1} + u_{0,2} + u_{0,0} - 4u_{0,1} = h^2 u_{0,1}^3$
5	$(1/2, 1/2)$	$u_{2,1} + u_{0,1} + u_{1,2} + u_{1,0} - 4u_{1,1} = h^2 u_{1,1}^3$

Taking advantage of symmetry and boundary conditions, it is clear that

$$u_1 = u_{0,0} \qquad u_2 = u_{1,0} = u_{-1,0} \qquad u_3 = u_{2,0}$$
$$u_4 = u_{0,1} = u_{0,-1} \quad u_5 = u_{1,1} = u_{-1,1} = u_{1,-1} \quad u_6 = u_{2,1}\,.$$

The simplified form of the finite difference equations is now

Node	Equation
1	$2u_2 + 2u_4 - 4u_1 = h^2 u_1^3$
2	$u_3 + u_1 + 2u_5 - 4u_2 = h^2 u_2^3$
4	$2u_5 + c + u_1 - 4u_4 = h^2 u_4^3$
5	$u_6 + u_4 + c + u_2 - 4u_5 = h^2 u_5^3$.

The treatment of the gradient boundary condition can be done in two ways.

Fictitious nodes Using this approach, it follows that $u_{3,0} = u_{1,0} = u_2$ and $u_{3,1} = u_{1,1} = u_5$. The partial differential equation applied at nodes 3 and 6 yields

Node	Equation		
3	$u_{3,0} + u_{1,0} + u_{2,1} + u_{2,-1} - 4u_{2,0} = h^2 u_{2,0}^3$	$\rightarrow$	$2u_2 + 2u_6 - 4u_3 = h^2 u_3^3$
6	$u_{3,1} + u_{1,1} + u_{2,2} + u_{2,0} - 4u_{2,1} = h^2 u_{2,1}^3$	$\rightarrow$	$2u_5 + c + u_3 - 4u_6 = h^2 u_6^3$.

The finite difference equations are now rewritten in iterative form by solving the equation derived from node k for u_k to get

Node	Equation	Gauss-Seidel Iteration
1	$u_1 = \frac{2u_2 + 2u_4}{4 + h^2 u_1^2}$	$u_1^{(k+1)} = \frac{2u_2^{(k)} + 2u_4^{(k)}}{4 + h^2 [u_1^{(k)}]^2}$
2	$u_2 = \frac{u_3 + u_1 + 2u_5}{4 + h^2 u_2^2}$	$u_2^{(k+1)} = \frac{u_3^{(k)} + u_1^{(k+1)} + 2u_5^{(k)}}{4 + h^2 [u_2^{(k)}]^2}$
3	$u_3 = \frac{2u_2 + 2u_6}{4 + h^2 u_3^2}$	$u_3^{(k+1)} = \frac{2u_2^{(k+1)} + 2u_6^{(k)}}{4 + h^2 [u_3^{(k)}]^2}$
4	$u_4 = \frac{2u_5 + c + u_1}{4 + h^2 u_4^2}$	$u_4^{(k+1)} = \frac{2u_5^{(k)} + c + u_1^{(k+1)}}{4 + h^2 [u_4^{(k)}]^2}$
5	$u_5 = \frac{u_6 + u_4 + c + u_2}{4 + h^2 u_5^2}$	$u_5^{(k+1)} = \frac{u_6^{(k)} + u_4^{(k+1)} + c + u_2^{(k+1)}}{4 + h^2 [u_5^{(k)}]^2}$
6	$u_6 = \frac{2u_5 + c + u_3}{4 + h^2 u_6^2}$	$u_6^{(k+1)} = \frac{2u_5^{(k+1)} + c + u_3^{(k+1)}}{4 + h^2 [u_6^{(k)}]^2}$.

Take at starting condition $u_1^{(0)} = u_2^{(0)} = u_3^{(0)} = u_4^{(0)} = u_5^{(0)} = u_6^{(0)} = c$.

Backward difference Using this approach, it follows that

Node	Equation		
3	$u_x(1,0) = \dfrac{u_{0,0} - 4u_{1,0} + 3u_{2,0}}{2h} = 0$	$\rightarrow$	$u_1 - 4u_2 + 3u_3 = 0$
6	$u_x(1,1/2) = \dfrac{u_{0,1} - 4u_{1,1} + 3u_{2,1}}{2h} = 0$	$\rightarrow$	$u_4 - 4u_5 + 3u_6 = 0\,.$

These equation are now used to eliminate u_3 and u_6 to obtain

$$\begin{aligned} h^2u_1^3 &= 2u_2 + 2u_4 - 4u_1 \\ h^2u_2^3 &= \frac{2}{3}u_1 + 2u_5 - \frac{8}{3}u_2 \\ h^2u_4^3 &= 2u_5 + c + u_1 - 4u_4 \\ h^2u_5^3 &= \frac{2}{5}\,u_4 + c + u_2 - \frac{8}{3}u_5\,. \end{aligned}$$

These equations are rewritten in fixed point form for u_1, u_2, u_4 and u_5.

Solution 27.

Each term in the finite difference scheme is replaced by its appropriate Taylor series. The component parts are:-

$$\begin{aligned} \frac{u_{i,j+1} - u_{i,j-1}}{2k} &= \frac{\partial u_{i,j}}{\partial t} + O(k^2) \\ \frac{u_{i,j} - u_{i,j-1}}{k} &= \frac{\partial u_{i,j}}{\partial t} - \frac{k}{2}\frac{\partial^2 u_{i,j}}{\partial t^2} + O(k^2) \\ \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} &= \frac{\partial^2 u_{i,j}}{\partial x^2} + \frac{h^2}{12}\frac{\partial^4 u_{i,j}}{\partial x^4} + O(h^4) \end{aligned}$$

The component parts are now assembled and lead to the conclusion that

$$\begin{aligned} &\theta\Big(\frac{u_{i,j+1} - u_{i,j-1}}{2k}\Big) + (1-\theta)\Big(\frac{u_{i,j} - u_{i,j-1}}{k}\Big) - \Big(\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2}\Big) \\ &= \theta\frac{\partial u_{i,j}}{\partial t} + (1-\theta)\Big(\frac{\partial u_{i,j}}{\partial t} - \frac{k}{2}\frac{\partial^2 u_{i,j}}{\partial t^2}\Big) - \Big(\frac{\partial^2 u_{i,j}}{\partial x^2} + \frac{h^2}{12}\frac{\partial^4 u_{i,j}}{\partial x^4}\Big) + O(k^2, h^4) \\ &= -\frac{k(1-\theta)}{2}\frac{\partial^2 u_{i,j}}{\partial t^2} - \frac{h^2}{12}\frac{\partial^4 u_{i,j}}{\partial x^4} + O(k^2, h^4) \end{aligned}$$

taking account of the fact that $u_t = u_{xx}$. Thus the truncation error is

$$-\frac{k(1-\theta)}{2}\frac{\partial^2 u}{\partial t^2} - \frac{h^2}{12}\frac{\partial^4 u}{\partial x^4} + O(k^2, h^4).$$

Furthermore, $u_{tt} = u_{txx} = u_{xxxx}$ and so this error terms further simplifies to

$$-\frac{h^2}{12}\left[6r(1-\theta) + 1\right]\frac{\partial^2 u}{\partial t^2} + O(k^2, h^4)\,, \qquad r = \frac{k}{h^2}.$$

Clearly it is beneficial to choose $\theta = 1 - 1/6r$.

Solution 28.

The explicit Euler representation of the partial differential equation is

$$u_{i,j+1} = r\,u_{i+1,j} + (1-2r)\,u_{i,j} + r\,u_{i-1,j}\,)$$

where $r = k/h^2$. In this application $k = 1/20$ and $h = 1/4$, and therefore $r = 4/5$ and the Euler algorithm becomes

$$u_{i,j+1} = \frac{1}{5}\Big[\,4u_{i+1,j} - 3u_{i,j} + 4u_{i-1,j}\,\Big].$$

The boundary conditions are $u_{0,j} = u_{4,j} = 0$. The system of differential equations to be solved is

$$\begin{aligned} u_{1,j+1} &= \frac{1}{5}\Big[\,4u_{2,j} - 3u_{1,j}\,\Big] \\ u_{2,j+1} &= \frac{1}{5}\Big[\,4u_{3,j} - 3u_{2,j} + 4u_{1,j}\,\Big] \\ u_{3,j+1} &= \frac{1}{5}\Big[\,-3u_{3,j} + 4u_{2,j}\,\Big] \end{aligned}$$

with initial conditions $u_{1,0} = 1/\sqrt{2}$, $u_{2,0} = 1$ and $u_{3,0} = 1/\sqrt{2}$. It is immediately obvious by subtracting the first and third equations that

$$u_{3,j+1} - u_{1,j+1} = -\frac{3}{5}\Big[\,u_{3,j} - u_{1,j}\,\Big].$$

Since the initial value of the difference is zero, then the difference remains zero, that is, $u_{1,j} = u_{3,j}$. Thus $u_{i,j}$ and $u_{2,j}$ satisfy

$$u_{1,j+1} = \frac{1}{5}\Big[\,4u_{2,j} - 3u_{1,j}\,\Big] \qquad u_{2,j+1} = \frac{1}{5}\Big[\,8u_{1,j} - 3u_{2,j}\,\Big]$$

with initial conditions $u_{1,0} = 1/\sqrt{2}$, $u_{2,0} = 1$. We are required to estimate $u_{2,2}$. First integration gives

$$u_{1,1} = \frac{1}{5}\Big[\,4 - 3/\sqrt{2}\,\Big] = \frac{1}{5\sqrt{2}}\Big[\,4\sqrt{2} - 3\,\Big] \qquad u_{2,1} = \frac{1}{5}\Big[\,4\sqrt{2} - 3\,\Big].$$

First integration gives

$$u_{1,2} = \frac{1}{5}\Big[\,4u_{2,1} - 3u_{1,1}\,\Big] = \frac{1}{25\sqrt{2}}\Big[\,4\sqrt{2} - 3\,\Big]^2 \qquad u_{2,2} = \frac{1}{5}\Big[\,8u_{1,1} - 3u_{2,1}\,\Big] = \frac{1}{25}\Big[\,4\sqrt{2} - 3\,\Big]^2.$$

Therefore $u_{1,2} \approx 0.1997$ (0.2635) and $u_{2,2} \approx 0.2824$ (0.3727). Exact solution is $u(x,t) = e^{-\pi^2 t}\sin\pi x$.

Solution 29.

(a) The Crank-Nicolson algorithm with $h = 1/4$ and $k = 0.1$ corresponds to the Courant number $r = 1.6$. The re-arranged Crank-Nicolson iterative scheme is therefore

$$-u_{i+1,j+1} + \frac{13}{4}\,u_{i,j+1} - u_{i-1,j+1} = u_{i+1,j} - \frac{3}{4}\,u_{i,j} + u_{i-1,j}\,.$$

The boundary conditions are $u_{0,j} = u_{4,j} = 0$ giving the simplified iterative algorithm

$$\begin{aligned} -u_{2,j+1} + \frac{13}{4}\, u_{1,j+1} &= u_{2,j} - \frac{3}{4}\, u_{1,j} \\ -u_{3,j+1} + \frac{13}{4}\, u_{2,j+1} - u_{1,j+1} &= u_{3,j} - \frac{3}{4}\, u_{2,j} + u_{1,j} \\ \frac{13}{4}\, u_{3,j+1} - u_{2,j+1} &= -\frac{3}{4}\, u_{3,j} + u_{2,j}\,. \end{aligned}$$

As in the previous example, the solution is symmetric about $x = 1/2$ so that $u_{1,j} = u_{3,j}$. With this additional simplification, it follows that

$$\left.\begin{aligned} \frac{13}{4}\, u_{1,j+1} - u_{2,j+1} &= -\frac{3}{4}\, u_{1,j} + u_{2,j} \\ -2u_{1,j+1} + \frac{13}{4}\, u_{2,j+1} &= 2u_{1,j} - \frac{3}{4}\, u_{2,j} \end{aligned}\right] \quad\rightarrow\quad \begin{aligned} u_{1,j+1} &= -\frac{7u_{1,j}}{137} + \frac{40u_{2,j}}{137} \\ u_{2,j+1} &= \frac{80u_{1,j}}{137} - \frac{7u_{2,j}}{137}\,. \end{aligned}$$

With initial conditions $u_{1,0} = 1/\sqrt{2}$ and $u_{2,0} = 1$, it is simple arithmetic to verify that $u_{1,1} = 0.2558$ (0.2635) and $u_{2,1} = 0.3618$ (0.3727).

(b) The Crank-Nicolson algorithm with $h = 1/4$ and $k = 0.025$ corresponds to the Courant number $r = 0.4$. The re-arranged Crank-Nicolson iterative scheme is therefore

$$-u_{i+1,j+1} + 7u_{i,j+1} - u_{i-1,j+1} = u_{i+1,j} + 3u_{i,j} + u_{i-1,j}\,.$$

The boundary conditions are $u_{0,j} = u_{4,j} = 0$ and give rise to the simplified iterative algorithm

$$\begin{aligned} -u_{2,j+1} + 7u_{1,j+1} &= u_{2,j} + 3u_{1,j} \\ -u_{3,j+1} + 7u_{2,j+1} - u_{1,j+1} &= u_{3,j} + 3u_{2,j} + u_{1,j} \\ 7u_{3,j+1} - u_{2,j+1} &= 3u_{3,j} + u_{2,j}\,. \end{aligned}$$

The symmetry of the solution about $x = 1/2$ means that $u_{1,j} = u_{3,j}$. With this additional simplification, it follows that

$$\left.\begin{aligned} 7u_{1,j+1} - u_{2,j+1} &= 3u_{1,j} + u_{2,j} \\ -2u_{1,j+1} + 7u_{2,j+1} &= 2u_{1,j} + 3u_{2,j} \end{aligned}\right] \quad\rightarrow\quad \begin{aligned} u_{1,j+1} &= \frac{23u_{1,j}}{47} + \frac{7u_{2,j}}{47} \\ u_{2,j+1} &= \frac{20u_{1,j}}{47} + \frac{23u_{2,j}}{47}\,. \end{aligned}$$

With initial conditions $u_{1,0} = 1/\sqrt{2}$ and $u_{2,0} = 1$, it is straight forward arithmetic to verify that $u_{1,1} = 0.49497$ and $u_{2,1} = 0.79026$. One further calculation gives $u_{2,2} = 0.59735$ (0.61050).

Solution 30.

The explicit Euler representation of the partial differential equation is

$$u_{i,j+1} - u_{i,j} = r\, x_i\, (\, u_{i+1,j} - 2u_{i,j} + u_{i-1,j}\,)$$

where $r = k/h^2$. This formulae may in turn be re-written in the explicit form

$$u_{i,j+1} = r\,x_i\,u_{i+1,j} + (1 - 2r\,x_i)u_{i,j} + r\,x_i u_{i-1,j}$$

The fictitious nodes procedure for treating the gradient boundary condition introduces a fictitious solution $u_{n+1,j}$ and removes this solution by eliminating it between the equations

$$\begin{aligned}
&\frac{\partial u(1/2,0)}{\partial x} = -\frac{u_{n,j}}{2} = \frac{u_{n+1,j} - u_{n-1,j}}{2h} \quad \rightarrow \quad u_{n+1,j} = u_{n-1,j} - hu_{n,j} \\
&\frac{\partial u(1/2,t)}{\partial t} = \frac{1}{2}\frac{\partial^2 u(1/2,t)}{\partial x^2} \quad \rightarrow \quad u_{n,j+1} = \frac{r}{2}\,u_{n+1,j} + (1-r)u_{n,j} + \frac{r}{2}u_{n-1,j}\,.
\end{aligned}$$

The result of this elimination is the condition

$$u_{n,j+1} = \Big(1 - r - \frac{rh}{2}\Big)u_{n,j} + ru_{n-1,j}\,.$$

The boundary condition at $x = 0$ is $u_{0,j} = 0$, and therefore the numerical problem to be solved consists of the equations

$$\begin{aligned}
i = 1 \qquad u_{1,j+1} &= (1 - 2rh)u_{1,j} + rh\,u_{2,j} \\
1 < i < n \quad u_{i,j+1} &= r\,x_i\,u_{i+1,j} + (1 - 2r\,x_i)u_{i,j} + r\,x_i u_{i-1,j} \\
i = n \qquad u_{n,j+1} &= \Big(1 - r - \frac{rh}{2}\Big)u_{n,j} + ru_{n-1,j}\,.
\end{aligned}$$

In this example $k = 0.005$ and $h = 1/10$, and therefore $r = 1/2$ and $n = 5$. The explicit Euler algorithm algorithm is now particularised to the equations

$$\begin{aligned}
u_{1,j+1} &= \frac{18u_{1,j} + u_{2,j}}{20} \\
u_{2,j+1} &= \frac{u_{1,j} + 8u_{2,j} + u_{3,j}}{10} \\
u_{3,j+1} &= \frac{3u_{2,j} + 14u_{3,j} + 3u_{4,j}}{20} \\
u_{4,j+1} &= \frac{u_{3,j} + 3u_{4,j} + u_{5,j}}{5} \\
u_{5,j+1} &= \frac{20u_{4,j} + 19u_{5,j}}{40}
\end{aligned}$$

which are iterated starting with

$$u_{1,0} = 0.09\,, \quad u_{2,0} = 0.16\,, \quad u_{3,0} = 0.21\,, \quad u_{4,0} = 0.24\,.$$

To determine whether or not the solution is well-behaved at $(1/2, 0)$, we check if the gradient boundary condition is satisfied there. Clearly $u(1/2, 0) = 0.35$ while the gradient is zero there. Clearly the gradient boundary condition is not instantaneously satisfied initially, and so we expect trouble with the numerical scheme.

Solution 31.

The general iterative scheme in the previous question is

$$\begin{aligned} i=1 \qquad u_{1,j+1} &= (1-2rh)u_{1,j} + rh\, u_{2,j} \\ 1<i<n \quad u_{i,j+1} &= r\,x_i\, u_{i+1,j} + (1-2r\,x_i)u_{i,j} + r\,x_i u_{i-1,j} \\ i=n \qquad u_{n,j+1} &= \Big(1-r-\frac{rh}{2}\Big)u_{n,j} + r u_{n-1,j}\,. \end{aligned}$$

The solution of this differential equation may therefore be written in the form $U_{j+1} = AU_j$ where the matrix A has form

$$\begin{bmatrix} (1-2rh) & rh & 0 & 0 & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & 0 & r\,x_i & (1-2r\,x_i) & r\,x_i & 0 & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & 0 & r & 1-r-\dfrac{rh}{2} \end{bmatrix}$$

We require the eigenvalues of this matrix to lie within the unit circle. The first Gershgorin circle is $|z-(1-2rh)| = rh$. This circle has centre on the x axis and the extremities of its diameter are at the points $(1-rh, 0)$ and $(1-3rh, 0)$. The second Gershgorin circle is $|z-(1-2rx_i)| = rx_i$. This circle also has centre on the x axis and the extremities of its diameter are at the points $(1, 0)$ and $(1-4rx_i, 0)$ where $0 < x_i < 1/2$. These two circles are contained within the unit circle centred on the origin provided

$$\left.\begin{array}{rcl} 1-3rh & > & -1 \\ 1-4rx_i & > & -1 \end{array}\right] \quad \rightarrow \quad \left.\begin{array}{rcl} 2 & > & 3rh \\ 1 & > & 2rx_i \end{array}\right] \quad \rightarrow \quad r<1$$

since $0 < x_i < 1/2$. The third Gershgorin circle is

$$\Big|\, z-\Big(1-r-\frac{rh}{2}\Big)\,\Big| = r$$

This circle also has centre on the x axis and its diameter extends over the interval

$$1-2r-rh/2 < x < 1-\frac{rh}{2} < 1\,.$$

Thus the third Gershgorin circle is entirely contained within the unit circle provided

$$1-2r-rh/2 > -1 \quad \rightarrow \quad r < \frac{4}{4+h}.$$

Solution 32.

Some useful Taylor series expansions of $u(x+h, t+k)$ are

$$\begin{aligned}
\frac{u_{i,j+1} - u_{i,j-1}}{2k} &= \frac{\partial u_{i,j}}{\partial t} + O(k^2) \\
\frac{u_{i,j+1} - u_{i,j}}{k} &= \frac{\partial u_{i,j}}{\partial t} + \frac{k}{2}\frac{\partial^2 u_{i,j}}{\partial t^2} + O(k^2) \\
\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} &= \frac{\partial^2 u_{i,j}}{\partial x^2} + \frac{h^2}{12}\frac{\partial^4 u_{i,j}}{\partial x^4} + O(h^4)
\end{aligned}$$

The left hand side of the iterative scheme may be rewritten

$$\begin{aligned}
&\frac{\alpha}{2k}\Big[\, 2u_{i,j+1} - (u_{i+1,j} + u_{i-1,j}) \,\Big] + \frac{(u_{i+1,j} - u_{i-1,j})}{2h} - f_{i,j} \\
&= \frac{\alpha}{2k}\Big[\, 2(u_{i,j+1} - u_{i,j}) - (u_{i+1,j} - 2u_{i,j} + u_{i-1,j}) \,\Big] + \frac{(u_{i+1,j} - u_{i-1,j})}{2h} - f_{i,j} \\
&= \frac{\alpha}{2k}\Big[\, 2k\frac{\partial u_{i,j}}{\partial t} + k^2\frac{\partial^2 u_{i,j}}{\partial t^2} - h^2\frac{\partial^2 u_{i,j}}{\partial x^2} - \frac{h^4}{12}\frac{\partial^4 u_{i,j}}{\partial x^4} + O(k^3, h^6) \,\Big] + \frac{\partial u_{i,j}}{\partial x} + O(h^2) - f_{i,j} \\
&= \alpha\,\frac{\partial u_{i,j}}{\partial t} + \frac{\partial u_{i,j}}{\partial x} - f_{i,j} + \frac{k\alpha}{2}\left[\, \frac{\partial^2 u_{i,j}}{\partial t^2} - \frac{h^2}{k^2}\,\frac{\partial^2 u_{i,j}}{\partial x^2} - \frac{h^4}{12k^2}\frac{\partial^4 u_{i,j}}{\partial x^4} \,\right] + O(k^2, h^6/k, h^2) \\
&= \frac{k\alpha}{2}\left[\, \frac{\partial^2 u_{i,j}}{\partial t^2} - \frac{h^2}{k^2}\,\frac{\partial^2 u_{i,j}}{\partial x^2} - \frac{h^4}{12k^2}\frac{\partial^4 u_{i,j}}{\partial x^4} \,\right] + O(k^2, h^6/k, h^2)
\end{aligned}$$

(a) Take $r = k/h$ then the remainder is

$$\frac{k\alpha}{2}\left[\, \frac{\partial^2 u_{i,j}}{\partial t^2} - \frac{1}{r^2}\,\frac{\partial^2 u_{i,j}}{\partial x^2} - \frac{h^2}{12r^2}\frac{\partial^4 u_{i,j}}{\partial x^4} \,\right] + O(k^2, h^5, h^2)$$

As $h \to 0$ and $k \to 0$ such that r is constant then the remainder also tends to zero and the scheme is consistent.

(b) Take $r = k/h^2$ then the remainder is

$$\frac{\alpha}{2}\left[\, k\frac{\partial^2 u_{i,j}}{\partial t^2} - \frac{1}{r}\,\frac{\partial^2 u_{i,j}}{\partial x^2} - \frac{h^2}{12r}\frac{\partial^4 u_{i,j}}{\partial x^4} \,\right] + O(k^2, h^4, h^2)$$

When $h \to 0$ and $k \to 0$ such that r is constant, the remainder no longer tends to zero and so the scheme is not consistent. The scheme now gives the solution to the equation

$$\alpha u_t + u_x - \frac{c\alpha}{2}u_{xx} - f(x,t) = 0$$

where c is a positive constant.

Solution 33.

The numerical scheme is

$$u_{i,j+1} - u_{i,j} = r\,\Big[\, \theta(u_{i-1,j+1} - 2u_{i,j+1} + u_{i+1,j+1}) + (1-\theta)(u_{i-1,j} - 2u_{i,j} + u_{i+1,j}) \,\Big]$$

in the usual notation where $\theta \in [0,1]$, $h = 1/n$ and $r = k/h^2$ is the Courant number. The stability of this numerical scheme is investigated by substituting

$$u_{p,q} = \lambda^q e^{i\alpha ph}$$

to obtain

$$\begin{aligned}\lambda^{q+1}e^{i\alpha ph} - \lambda^q e^{i\alpha ph} &= r\Big[\theta(\lambda^{q+1}e^{i\alpha(p-1)h} - 2\lambda^{q+1}e^{i\alpha ph} + \lambda^{q+1}e^{i\alpha(p+1)h}) \\ &\qquad + (1-\theta)(\lambda^q e^{i\alpha(p-1)h} - 2\lambda^q e^{i\alpha ph} + \lambda^q e^{i\alpha(p+1)h})\Big].\end{aligned}$$

Cancellation of $\lambda^q e^{i\alpha ph}$ now simplifies the previous expression to give

$$\begin{aligned}\lambda - 1 &= r\Big[\theta\lambda(e^{-i\alpha h} - 2 + e^{i\alpha h}) + (1-\theta)(e^{-i\alpha h} - 2 + e^{i\alpha h})\Big] \\ &= r\,(1 - \theta + \lambda\theta)\,(e^{-i\alpha h} - 2 + e^{i\alpha h}) \\ &= 2r\,(1 - \theta + \lambda\theta)\,(\cos\alpha h - 1)\,.\end{aligned}$$

This is a linear equation for λ with solution

$$\lambda = \frac{1 - 2r\,(1-\theta)\,(1-\cos\alpha h)}{1 + 2r\,\theta\,(1-\cos\alpha h)}\,.$$

The denominator of this fraction is always positive and so $|\lambda| < 1$ provided

$$\begin{aligned}& -1 - 2r\,\theta\,(1-\cos\alpha h) < 1 - 2r\,(1-\theta)\,(1-\cos\alpha h) < 1 + 2r\,\theta\,(1-\cos\alpha h) \\ \rightarrow\quad & -2 - 2r\,\theta\,(1-\cos\alpha h) < -2r\,(1-\theta)\,(1-\cos\alpha h) < 2r\,\theta\,(1-\cos\alpha h) \\ \rightarrow\quad & -1 - r\,\theta\,(1-\cos\alpha h) < -r\,(1-\theta)\,(1-\cos\alpha h) < r\,\theta\,(1-\cos\alpha h) \\ \rightarrow\quad & -1 + r\,(1-2\theta)\,(1-\cos\alpha h) < 0 < r\,(1-\cos\alpha h)\end{aligned}$$

The right hand inequality simply asserts that r must be positive and so this inequality is satisfied automatically. The second inequality asserts that

$$r < \frac{1}{2(1-2\theta)\,\sin^2(\alpha h/2)}$$

In particular, $\theta \in [0, 1/2)$ to ensure that $r > 0$. Subject to this constraint on θ, r must be less than the minimum upper bound as α varies, that is,

$$0 < r < \frac{1}{2(1-2\theta)}\,, \qquad \theta \in [0, 1/2)\,.$$

Solution 34.

Let T denote the tri-diagonal matrix of the question, then

$$T = \begin{bmatrix} a & b & 0 & 0 & \cdots & \cdots & 0 \\ c & a & b & 0 & \cdots & \cdots & 0 \\ 0 & c & a & b & 0 & \cdots & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & b \\ 0 & \cdots & \cdots & \cdots & 0 & c & a \end{bmatrix}$$

and let the column vector X_k be defined by the formula

$$X_k = \Big[\Big(\frac{c}{b}\Big)^{1/2} \sin\theta_k, \Big(\frac{c}{b}\Big) \sin 2\theta_k, \cdots, \Big(\frac{c}{b}\Big)^{j/2} \sin j\theta_k, \cdots, \Big(\frac{c}{b}\Big)^{n/2} \sin n\theta_k \Big]^{\mathrm{T}} .$$

To establish the required result, it is enough to demonstrate that $TX_k = \lambda_k X_k$ for any suitable integer k.

The computation of the first and last entries of TX_k require special attention.

First row of TX_k gives the matrix product

$$\begin{aligned} a\Big(\frac{c}{b}\Big)^{1/2} \sin\theta_k + b\Big(\frac{c}{b}\Big) \sin 2\theta_k &= \Big(\frac{c}{b}\Big)^{1/2} \Big[a\sin\theta_k + 2b\Big(\frac{c}{b}\Big)^{1/2} \sin\theta_k \cos\theta_k \Big] \\ &= \Big(\frac{c}{b}\Big)^{1/2} \sin\theta_k \Big[a + 2\sqrt{bc}\, \cos\theta_k \Big] \\ &= \lambda_k \Big(\frac{c}{b}\Big)^{1/2} \sin\theta_k \,. \end{aligned}$$

Intermediate rows of TX_k, say j^{th} row, give the matrix product

$$\begin{aligned} & c\Big(\frac{c}{b}\Big)^{(j-1)/2} \sin(j-1)\theta_k + a\Big(\frac{c}{b}\Big)^{j/2} \sin j\theta_k + b\Big(\frac{c}{b}\Big)^{(j+1)/2} \sin(j+1)\theta_k \\ &= \Big(\frac{c}{b}\Big)^{j/2} \Big[c\Big(\frac{c}{b}\Big)^{-1/2} \sin(j-1)\theta_k + a\sin j\theta_k + b\Big(\frac{c}{b}\Big)^{1/2} \sin(j+1)\theta_k \Big] \\ &= \Big(\frac{c}{b}\Big)^{j/2} \Big[\sqrt{bc}\, \sin(j-1)\theta_k + a\sin j\theta_k + \sqrt{bc}\, \sin(j+1)\theta_k \Big] \\ &= \Big(\frac{c}{b}\Big)^{j/2} \Big[2\sqrt{bc}\, \sin j\theta_k \cos\theta_k + a\sin j\theta_k \Big] \\ &= \Big(\frac{c}{b}\Big)^{j/2} \sin j\theta_k \Big[a + 2\sqrt{bc}\, \cos\theta_k \Big] \\ &= \lambda_k \Big(\frac{c}{b}\Big)^{j/2} \sin j\theta_k \,. \end{aligned}$$

Last row of TX_k gives the matrix product

$$\begin{aligned} & c\Big(\frac{c}{b}\Big)^{(n-1)/2}\sin(n-1)\theta_k + a\Big(\frac{c}{b}\Big)^{n/2}\sin n\theta_k \\ &= \Big(\frac{c}{b}\Big)^{n/2}\Big[\, c\Big(\frac{c}{b}\Big)^{-1/2}\sin(n-1)\theta_k + a\sin n\theta_k\,\Big] \\ &= \Big(\frac{c}{b}\Big)^{n/2}\Big[\, \sqrt{bc}\,\sin(n-1)\theta_k + a\sin n\theta_k\,\Big]\,. \end{aligned}$$

The identity $\sin(n-1)\theta_k + \sin(n+1)\theta_k = 2\sin n\theta_k\cos\theta_k$ implies that $\sin(n-1)\theta_k = 2\sin n\theta_k\cos\theta_k$ in view on the definition of θ_k. Therefore,

$$c\Big(\frac{c}{b}\Big)^{(n-1)/2}\sin(n-1)\theta_k + a\Big(\frac{c}{b}\Big)^{n/2}\sin n\theta_k = \lambda_k\left(\frac{c}{b}\right)^{n/2}\sin n\theta_k\,.$$

In conclusion λ_k is an eigenvalue of T with eigenvector X_k.

Solution 35.

The numerical scheme is

$$u_{i,j+1} - u_{i,j} = r\alpha(u_{i-1,j+1} - 2u_{i,j+1} + u_{i+1,j+1})$$

in the usual notation where $\theta \in [0,1]$, $h = 1/n$ and $r = k/h^2$ is the Courant number.

(a) The Fourier method investigates the stability of the numerical scheme by substituting

$$u_{p,q} = \lambda^q e^{i\beta ph}$$

in the scheme to obtain

$$\lambda^{q+1}e^{i\beta ph} - \lambda^q e^{i\beta ph} = r\alpha(\lambda^{q+1}e^{i\beta(p-1)h} - 2\lambda^{q+1}e^{i\beta ph} + \lambda^{q+1}e^{i\beta(p+1)h})\,.$$

After cancellation of $\lambda^q e^{i\beta ph}$ this equation simplifies to

$$\lambda - 1 = r\alpha\lambda(e^{-i\beta h} - 2 + e^{i\beta h}) = 2r\alpha\lambda(\cos\beta h - 1) = -4r\alpha\lambda\sin^2\beta h/2\,.$$

Thus it is seen that

$$\lambda = \frac{1}{1 + 4r\alpha\sin^2\beta h/2}$$

Since $0 < \lambda < 1$ for all choices of r and positive α, then the numerical algorithm is unconditionally stable.

(b) The Matrix method investigates the stability of the numerical scheme

$$u_{i,j+1} - u_{i,j} = r\alpha(u_{i-1,j+1} - 2u_{i,j+1} + u_{i+1,j+1})$$

by rewriting it in the form

$$-r\alpha u_{i-1,j+1} + (1 + 2r\alpha)u_{i,j+1} - r\alpha u_{i+1,j+1} = u_{i,j}\,.$$

The boundary conditions are $u_{0,j} = u_{n,j} = 0$ and therefore the equations to be solved are

$$\begin{aligned} i = 1 \qquad & (1+2r\alpha)u_{1,j+1} - r\alpha u_{2,j+1} &=& \; u_{1,j} \\ 1 < i < n-1 \quad & -r\alpha u_{i-1,j+1} + (1+2r\alpha)u_{i,j+1} - r\alpha u_{i+1,j+1} &=& \; u_{i,j} \\ i = n-1 \qquad & -r\alpha u_{n-2,j+1} + (1+2r\alpha)u_{n-2,j+1} &=& \; u_{i,j}\,. \end{aligned}$$

There equations can be assembled into the tri-diagonal form $TU_{j+1} = U_j$ where T is the tri-diagonal matrix

$$T = \begin{bmatrix} (1+2r\alpha) & -r\alpha & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ -r\alpha & (1+2r\alpha) & -r\alpha & 0 & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & 0 & -r\alpha & (1+2r\alpha) & -r\alpha & 0 & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & -r\alpha & (1+2r\alpha) \end{bmatrix}$$

The numerical scheme $TU_{j+1} = U_j$ will be stable provided the eigenvalues of T^{-1} all lie within the unit circle. However, if the eigenvalues of T^{-1} all lie within the unit circle, then the eigenvalues of T are required to lie outside the unit circle to guarantee stability. The eigenvalues of T are

$$\lambda_i = 1 + 2\alpha r + 2\,\alpha r\,\cos i\pi/n = 1 + 2\alpha r(1+\cos i\pi/n) = 1 + 4\alpha r \sin^2(i\pi/2n) > 1$$

and so the previous condition is satisfied and the numerical scheme is unconditionally stable.

Solution 36.

(a) The Euler algorithm is

$$u_{i,j+1} - r\,u_{i+1,j} - (1-2r)\,u_{i,j} - r\,u_{i-1,j} = 0\,.$$

To establish the consistency of this scheme, it is convenient to re-express the scheme in the form

$$u_{i,j+1} - u_{i,j} - r\,(u_{i+1,j} - 2\,u_{i,j} + u_{i-1,j}) = 0\,.$$

The component quantities appearing in this algorithm are now replaced by their Taylor series forms

$$\begin{aligned} u_{i,j+1} - u_{i,j} &= k\frac{\partial u_{i,j}}{\partial t} + \frac{k^2}{2}\frac{\partial^2 u_{i,j}}{\partial t^2} + O(k^2) \\ u_{i+1,j} - 2\,u_{i,j} + u_{i-1,j} &= h^2\frac{\partial^2 u_{i,j}}{\partial x^2} + \frac{h^4}{12}\frac{\partial^4 u_{i,j}}{\partial x^4} + O(h^6) \end{aligned}$$

to obtain

$$\begin{aligned} & u_{i,j+1} - u_{i,j} - r\,(u_{i+1,j} - 2\,u_{i,j} + u_{i-1,j}) \\ & \quad = k\frac{\partial u_{i,j}}{\partial t} + \frac{k^2}{2}\frac{\partial^2 u_{i,j}}{\partial t^2} - r\Big(h^2\frac{\partial^2 u_{i,j}}{\partial x^2} + \frac{h^4}{12}\frac{\partial^4 u_{i,j}}{\partial x^4}\Big) + O(k^2, rh^6) \\ & \quad = k\Big(\frac{\partial u_{i,j}}{\partial t} - \frac{\partial^2 u_{i,j}}{\partial x^2}\Big) + \frac{k^2}{2}\frac{\partial^2 u_{i,j}}{\partial t^2} - \frac{kh^2}{12}\frac{\partial^4 u_{i,j}}{\partial x^4} + O(k^2, kh^4) \\ & \quad = \frac{k^2}{2}\frac{\partial^2 u_{i,j}}{\partial t^2} - \frac{kh^2}{12}\frac{\partial^4 u_{i,j}}{\partial x^4} + O(k^2, kh^4)\,. \end{aligned}$$

This tends towards zero as $h \to 0^+$ and $k \to 0^+$ and so the Euler algorithm is consistent. Note again that the choice $r = 1/6$ is particularly beneficial in the case of the diffusion equation.

(b) The Richardson algorithm is

$$u_{i,j+1} - u_{i,j-1} - 2r\,(u_{i+1,j} - 2u_{i,j} + u_{i-1,j}\,) = 0\,.$$

The component quantities appearing in this algorithm are now replaced by their Taylor series forms

$$\begin{aligned} u_{i,j+1} - u_{i,j-1} &= 2k\frac{\partial u_{i,j}}{\partial t} + \frac{k^3}{3}\frac{\partial^2 u_{i,j}}{\partial t^2} + O(k^5) \\ u_{i+1,j} - 2\,u_{i,j} + u_{i-1,j} &= h^2\frac{\partial^2 u_{i,j}}{\partial x^2} + \frac{h^4}{12}\frac{\partial^4 u_{i,j}}{\partial x^4} + O(h^6) \end{aligned}$$

to obtain

$$\begin{aligned} & u_{i,j+1} - u_{i,j-1} - 2r\,(u_{i+1,j} - 2u_{i,j} + u_{i-1,j}) \\ & \quad = 2k\frac{\partial u_{i,j}}{\partial t} - 2r\Big(h^2\frac{\partial^2 u_{i,j}}{\partial x^2} + \frac{h^4}{12}\frac{\partial^4 u_{i,j}}{\partial x^4}\Big) + O(k^3, rh^6) \\ & \quad = 2k\Big(\frac{\partial u_{i,j}}{\partial t} - \frac{\partial^2 u_{i,j}}{\partial x^2}\Big) - \frac{kh^2}{6}\frac{\partial^4 u_{i,j}}{\partial x^4} + O(k^3, kh^4) \\ & \quad = -\frac{kh^2}{6}\frac{\partial^4 u_{i,j}}{\partial x^4} + O(k^3, kh^4)\,. \end{aligned}$$

This tends towards zero as $h \to 0^+$ and $k \to 0^+$ and so the Richardson algorithm is consistent.

Solution 37.

The stability of the 2-D ADI algorithm is investigated by the substitution

$$u_{p,q}^n = \lambda^n e^{i(\alpha p + \beta q)h}\,.$$

The calculation is based on the auxiliary observation that

$$\begin{aligned} \delta_x^2\,u &= (e^{i\alpha h} - 2 + e^{-i\alpha h})\,u = 2(\cos\alpha h - 1)\,u = -4(\sin^2\alpha h/2)\,u\,, \\ \delta_y^2\,u &= (e^{i\beta h} - 2 + e^{-i\beta h})\,u = 2(\cos\beta h - 1)\,u = -4(\sin^2\beta h/2)\,u\,. \end{aligned}$$

The quantity $\bar{u}^{n+1}$ is first eliminated between the equations

$$\begin{aligned}(1-r\delta_x^2)\bar{u}^{n+1} &= (1+r\delta_y^2)\,u^n,\\ (1-r\delta_y^2)u^{n+1} &= \bar{u}^{n+1}-r\delta_y^2u^n\end{aligned}$$

by applying the operator $(1-r\delta_x^2)$ to the second equation and then taking advantage of the first equation. The result of this procedure is

$$(1-r\delta_x^2)(1-r\delta_y^2)u^{n+1}=(1+r\delta_y^2)\,u^n-r(1-r\delta_x^2)\delta_y^2u^n\,.$$

Taking account of the auxiliary observations,

$$\begin{aligned}\lambda(1+4r\sin^2(\alpha h/2))(1+4r\sin^2(\beta h/2)) &= (1-4r\sin^2(\beta h/2))+4r(1+4r\sin^2(\alpha h/2))(\sin^2\beta h/2)\\ &= 1+16r^2\sin^2(\alpha h/2)\sin^2(\beta h/2)\,.\end{aligned}$$

Thus we see that

$$0<\lambda=\frac{1+16r^2\sin^2(\alpha h/2)\sin^2(\beta h/2)}{1+4r\sin^2(\alpha h/2)+4r\sin^2(\beta h/2)+16r^2\sin^2(\alpha h/2)\sin^2(\beta h/2)}<1\,.$$

Thus the 2-D ADI method yields is unconditionally stable.

Solution 38.

The the 3-D ADI method is given by the iterative scheme

$$\begin{aligned}(1-r\delta_x^2)u^{n+1} &= (1+r\delta_y^2+r\delta_z^2)u^n,\\ (1-r\delta_y^2)u^{n+2} &= (1+r\delta_x^2+r\delta_z^2)u^{n+1},\\ (1-r\delta_z^2)u^{n+3} &= (1+r\delta_x^2+r\delta_y^2)u^{n+2}\,.\end{aligned}$$

The quantity u^{n+1} is first eliminated by multiplying the second equation by the operator $(1-r\delta_x^2)$ and then using the first equation. The result of this procedure is

$$(1-r\delta_x^2)(1-r\delta_y^2)u^{n+2}=(1+r\delta_x^2+r\delta_z^2)(1-r\delta_x^2)u^{n+1}=(1+r\delta_x^2+r\delta_z^2)(1+r\delta_y^2+r\delta_z^2)u^n\,.$$

The quantity u^{n+2} is now eliminated by multiplying the third equation by the operator $(1-r\delta_x^2)(1-r\delta_y^2)$ and then using the equation above. The result of this procedure is

$$(1-r\delta_x^2)(1-r\delta_y^2)(1-r\delta_z^2)u^{n+3}=(1+r\delta_x^2+r\delta_y^2)(1+r\delta_x^2+r\delta_z^2)(1+r\delta_y^2+r\delta_z^2)u^n\,.$$

The stability of the 3-D ADI algorithm is investigated by the substitution

$$u_{p,q}^n=\lambda^n e^{i(\alpha p+\beta q+\gamma s)h}\,.$$

The calculation is based on the auxiliary observation that

$$\begin{aligned}
\delta_x^2\, u &= (e^{i\alpha h} - 2 + e^{-i\alpha h})\, u = 2(\cos\alpha h - 1)\, u = -4\sin^2(\alpha h/2)\, u\,, \\
\delta_y^2\, u &= (e^{i\beta h} - 2 + e^{-i\beta h})\, u = 2(\cos\beta h - 1)\, u = -4\sin^2(\beta h/2)\, u\,, \\
\delta_z^2\, u &= (e^{i\gamma h} - 2 + e^{-i\gamma h})\, u = 2(\cos\gamma h - 1)\, u = -4\sin^2(\gamma h/2)\, u\,.
\end{aligned}$$

As in the previous question, it is straight forward algebra to confirm that λ satisfies

$$\lambda^3 = \frac{[1 - 4r(\sin^2(\alpha h/2) + \sin^2(\beta h/2))][1 - 4r(\sin^2(\alpha h/2) + \sin^2(\gamma h/2)][1 - 4r(\sin^2(\beta h/2) + \sin^2(\gamma h/2)]}{(1 + 4r\sin^2(\alpha h/2))(1 + 4r\sin^2(\beta h/2))(1 + 4r\sin^2(\gamma h/2))}\,.$$

In order to facilitate the discussion of λ^3, write

$$x = 4r\sin^2(\alpha h/2)\,, \qquad y = 4r\sin^2(\beta h/2)\,, \qquad z = 4r\sin^2(\gamma h/2)$$

so that

$$\lambda^3 = \frac{[1 - x - y][1 - x - z][1 - y - z]}{[1 + x][1 + y][1 + z]}\,.$$

By symmetry, the minimum/maximum values of λ^3 occur where $x = y = z$. In this case

$$\lambda^3 = \Big(\frac{1 - 2x}{1 + x}\Big)^3\,,$$

and this value will lie within $|\lambda| < 1$ provided

$$-1 < \frac{1 - 2x}{1 + x} < 1\,, \quad \rightarrow \quad -1 - x < 1 - 2x < 1 + x\,, \quad \rightarrow \quad -2 + x < 0 < 3x\,.$$

Since $x \geq 0$ by construction then we require $4r\sin^2(\alpha h/2) < 2$ for all choices of α. Thus $r \leq 1/2$ to guarantee stability.

Solution 39.

The central difference numerical scheme for the solution of the diffusion equation is

$$\frac{du_k}{dt} = \frac{1}{h^2}\Big[\, u_{k+1} - 2u_k + u_{k-1}\,\Big] + f_k(t) + O(h^2)\,, \qquad k = 1, 2, \cdots, n-1$$

where $f_k(t) = f(kh, t)$. The solution at $x = 0$ is given by the boundary condition $u_0(t) = 0$, and therefore it remains to construct the differential equation for $u_n(t)$ using the boundary condition. In the fictitious nodes procedure, this equation is determined by eliminating the fictitious solution $u_{n+1}(t)$ between the equations

$$\begin{aligned}
\frac{du_n}{dt} &= \frac{1}{h^2}\Big[\, u_{n+1} - 2u_n + u_{n-1}\,\Big] + f_n(t) + O(h^2) \\
\frac{u_{n+1} - u_{n-1}}{2h} &= \frac{\partial u(x_n, t)}{\partial x} + \frac{h^2}{6}\,\frac{\partial^3 u(x_n, t)}{\partial x^3} + O(h^4) = g(t) + \frac{h^2}{6}\,\frac{\partial^3 u(x_n, t)}{\partial x^3} + O(h^4)\,.
\end{aligned} \tag{6.7}$$

The second equation yields

$$u_{n+1} = u_{n-1} + 2h\frac{\partial u(x_n,t)}{\partial x} + O(h^3) = u_{n-1} + 2hg(t) + \frac{h^3}{3}\frac{\partial^3 u(x_n,t)}{\partial x^3} + O(h^5)$$

and when this identity is substituted into the first of equation (6.7), the result is that $u_n(t)$ satisfies

$$\begin{aligned}\frac{du_n}{dt} &= \frac{1}{h^2}\Big[\,2u_{n-1} + 2hg(t) + \frac{h^3}{3}\frac{\partial^3 u(x_n,t)}{\partial x^3} + O(h^5) - 2u_n\,\Big] + f_n(t) + O(h^2) \\ &= \frac{2(u_{n-1}-u_n)}{h^2} + \frac{2g(t)}{h} + f_n(t) + \frac{h}{3}\frac{\partial^3 u(x_n,t)}{\partial x^3} + O(h^3)\,.\end{aligned}$$

Thus the differential equation satisfied by u_n is only $O(h)$ accurate and so the solution of the finite difference equations in this case are $O(h)$ accurate.

Clearly $O(h^3)$ accuracy is recovered whenever $\partial^3 u(x_n,t)/\partial x^3 = 0$. It follows directly from the differential equation that

$$\frac{\partial^3 u(x_n,t)}{\partial x^3} = \frac{\partial}{\partial x}\Big(\frac{\partial u(x_n,t)}{\partial t} - f(x,t)\Big) = \frac{\partial}{\partial t}\Big(\frac{\partial u(x_n,t)}{\partial x}\Big) - \frac{\partial f(x_n,t)}{\partial x} = \frac{dg}{dt} - \frac{\partial f(x_n,t)}{\partial x}\,.$$

Thus $O(h^3)$ accuracy is recovered provided

$$\frac{dg}{dt} = \frac{\partial f(x_n,t)}{\partial x}$$